



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

**Escola Tècnica Superior d'Enginyeria
de Telecomunicació de Barcelona**

Discrete time control of a Push-Pull power converter

A Master's Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Jaume Balcells Ortega

In partial fulfilment

of the requirements for the degree of

MASTER IN ELECTRONIC ENGINEERING

Advisor: Domingo Biel Solé

Barcelona. June 27, 2018

Agraïments

Aquesta tesi de màster no seria possible sense l'ajuda del Jaume i l'Ana, els meus pares, gràcies per fer-me suport en totes les decisions que he pres, estic just on volia ser gràcies a això. Tampoc seria possible sense la meva germana Ingrid, la qual sempre hem dona els millors consells.

Alicia, moltes gràcies per ser la millor ajuda durant tots aquets dies. Als amics, no us preocupeu, torno a estar aquí !

Al meu tutor Domingo Biel Solé, merci per tot el suport tècnic i anímic prestat, en especial per fer divertits aquells moments més feixucs. També vull apreciar l'ajuda del professor Rafel Cardoner.

Durant la tesis he treballat a l'empresa Premium, fabricant d'electrònica de potència. Ha sigut un plaer rebre tot el coneixement i suport moral de tots els companys, en especial del Roger Ferrer, Manuel Camacho, Albert Solé, Jordi Bècares i del Jordi Salvador.

Moltes gràcies a tots els que heu fet possible aquesta tesi final de màster.

Acknowledgments

This master thesis will not be possible without all the efforts that Jaume and Ana have done, my parents, thanks for supporting all my decisions because I am just where I want to be. It will not be possible without my sister Ingrid, who always give me the best advises.

Alicia, thanks for being the best support for these days. To my worried friends just say, don't worry I am back again!

To my advisor Domingo Biel Solé, thanks for all the technical and empowering support received, specially for making funny all those moments of hard work. Also appreciating the advises from Rafel Cardoner.

During this time I was working in Premium company, a power conversion custom solutions provider from Barcelona. It was a pleasure to receive such appreciable knowledge and support from all the co-workers, in special mention to Roger Ferrer, Manuel Camacho, Albert Solé, Jordi Bècares and Jordi Salvador.

Thanks you very much to all who made possible this final master thesis happen.

Abstract

We live a world where the wealth can be measured with time, which is constantly changing and evolving. In this world electronic devices must be adapted for the pace of life, for those reasons digital control techniques become an important topic in power electronics since 1980s.

Using digital controllers it is possible to speed-up the design time, add more complex control algorithms, increase flexibility or introduce data monitoring and create power converters capable to interact with its environmental digital devices among other benefits.

The main objective of this thesis is to apply digital control techniques in a push-pull DC/DC power converter for improve its performance. It has been done with a commercial power converter from Premium S.A company and with C2000 TI DSP family.

Push-pull topology is an ideal solution for isolated DC/DC power converters, specially for low input voltages and for medium power conversion stages, less than 1000[W].

It has been compared continuous-time and discrete-time controller implementation concluding that it is possible to have equal or even better regulation results with digital controllers, while discrete ones can make power converters smarter.

Following chapters will describe how to model the push-pull converter, design several compensation algorithms, simulate its performance using Matlab[®] and PSIM[®] and finally implement the best solution in order to obtain experimental results.

Contents

1	Project definition	3
1.1	Objectives	3
1.2	Tasks	4
2	Introduction to digital power solution	5
2.1	Why digital?	5
2.2	Real examples	6
2.2.1	Cinergia case	6
2.2.2	D3 Engineering case	7
2.3	Power converter selected	8
3	Converter modeling	10
3.1	Push-pull converter schematic and state space equations	11
3.2	State space averaged model	14
3.3	Transfer functions	15
3.4	Steady State Analysis	17
4	Voltage Mode Control Design	21
4.1	Uncompensated closed-loop response	22
4.2	Proportional Integral compensator	24
4.3	Lead compensator	26
4.4	PI+Lead compensator	28
4.5	Proportional Integral Derivative compensator	31
4.6	PID vs PI+Lead	32
5	Simulations	35
5.1	Converter model	35
5.2	Continuous-time domain	36
5.2.1	Load disturbance	38
5.2.2	Line disturbance	38
5.2.3	Reference tracking	39
5.2.4	Pulse width modulation effects	40
5.3	Discrete-time domain	43
5.3.1	ADC	44
5.3.2	Digital PWM	45
5.3.3	Digital Compensator	46
5.3.4	Simulink results	49

5.4	PSIM Simulations	52
5.4.1	Compensation law	53
5.4.2	Simulation scenario	54
5.4.3	Digital PI+Lead controller	55
5.4.4	Digital PID controller	56
6	Implementation	58
6.1	Firmware	58
6.1.1	C2000 Launchpad	58
6.1.2	Program code	60
6.2	Hardware	62
6.2.1	Compensator	62
6.2.2	MOSFET Driver	63
6.2.3	Sensing	65
7	Experimental Results	70
7.1	Bench-test definition	70
7.2	Load regulation	71
7.3	Line regulation	72
7.4	Efficiency	73
7.5	Ripple and Noise	74
7.6	Transient response	75
7.7	Soft-start	77
8	Budget	80
8.1	Design, prototype and implementation costs	80
8.2	Prototype BOM	81
8.3	Financial viability analysis	82
9	Conclusions and future works	83
9.1	Conclusions	83
9.2	Future works	84
10	Annexes	86
10.1	Matlab files	86
10.2	PSIM controllers DLLs	93
10.3	Firmware code	96
10.4	Hardware Schematic	105

Chapter 1

Project definition

1.1 Objectives

Making things smarter has become the rule of thumb in modern engineering, this intelligence normally is followed by a computing system which processes the information and sends it to the digital world domain. Power conversion systems are not apart from all these conceptions, and here is where digital power solutions became really interesting.

This work applies digital control techniques in a push-pull power conversion system and in order to reach this goal the following objectives are introduced:

- Learn and understand about push-pull converter topology, being able to define its behavior with mathematical expressions and taking special attention about the control techniques usually adopted.
- Analyze the differences between analog and digital control technique implementation, learn about the main benefits and drawbacks exposing some conclusions about that.
- Design and compare several compensation algorithms for control the power stage, at least obtain two compensation expressions. Studying the perturbations rejection given for each of them and selecting the best option.
- Learn about design and simulation software tools usually used in control theory, specially from Matlab[®], Simulink[®] and PSIM[®].
- Study and explore existing development platforms and solutions in digital power applications, selecting one from the different possibilities in order to match the design requirements. With selected option, analyze its implementation boundaries.

- Take a commercial push-pull power converter operating with analog control and implement it digitally. Modify the needed parts, testing and making possible to obtain at least one feedback loop running with a digital controller.
- Define some laboratory tests in order to compare the results obtained in the implementation with the simulations and also with the manufacturer specifications.
- Analyze and obtain the analog controller expression from selected commercial converter, implement it in digital domain and compare its performances.
- Professional text editing resources like Latex code based give very good results in technical text edition, it is wanted to learn about this topic by redacting the memory.

1.2 Tasks

In order to reach this objectives in a structured way, a set of tasks are being defined:

- Project definition.
- Converter modeling.
- Compensator design.
- Converter hardware modifications design.
- Simulations.
- Implement digital controller.
- Implement hardware modifications.
- Obtain experimental results.
- Documentation.

Chapter 2

Introduction to digital power solution

2.1 Why digital?

Power electronics control came from classical analog techniques and implementation, which in essence compensate the error between sensed and reference signal continuously. One can think that a sampled version of the compensation technique can not give better results and that have no sense to take digital control, but in fact, it has in the modern engineering systems.

Digital control solutions are not a recent discovery, it debuted in motion control and UPS field in early 80's but it is becoming trending due to its multiple benefits;

Flexibility. Provably the most important benefit, while analog control is implemented by hardware, digital is done by firmware or software what makes more easy to change and adapt multiple solutions using the same hardware. Moreover, to implement more complex algorithms in digital control only needs to compile a new code while analog will need to redesign hardware with more complex structures.

Functionality. By using a digital processing system it is possible to include skills like monitoring, low-power modes, parameter adjustability, improved fault containment strategy, machine learning, load share or converter parallelization among others.

Accuracy. Compensator is defined by poles and zeros, analog implementations use capacitors that more often has high tolerance values, while digital controllers use 12-bits or more resolution to describe its value. Tolerances, aging, temperature effects, drifts or offsets contribute to have more accurate digital solutions.

Speed. Technology improvements made possible to use higher frequency clock signals in digital control solutions, which make possible to sample and compute signals similarly

to analog systems, even for high switching frequency converters requirements.

Cost. Very large-scale integration (VLSI) technology is continuously evolving which makes possible better and cheaper manufacturing process, giving more reliable IC solutions at low cost. Integrating the solution with a System on Chip can increase converters switching frequency and reduce power topology inductive component costs.

After those reasons it seems that digital is the best in the world, but obviously as every technology have some drawbacks and limitations. The main drawback is the processor clock frequency because it affects to the maximum affordable converter switching frequency, ADC-DAC speeds and compensation law bandwidth. Another important issue is the quantization errors, digital resolution is finite while analog not.

2.2 Real examples

Next section presents some applications where digital control is the key to success.

2.2.1 Cinergia case

Cinergia is a company focused in power electronics control. This is a very interesting case because with a single hardware topology they can offer multiple solutions with the same controller and hardware platforms.

They have developed in collaboration with Salicru company a control platform that manage a bidirectional three phase converter, it is known as a Back-to-Back or 4 quadrants topology because it can move power from one side to the other depending on the control guidelines. In fact, they can offer AC and DC power solutions with the same hardware parts, only changing some filtering components and the control firmware. Figure 2.1 shows a DC power supply solution.

Cinergia has an AC-3phase regenerative electronic loads for high power ranges while other laboratory instrument providers need huge and costly solutions to provide the same solution.

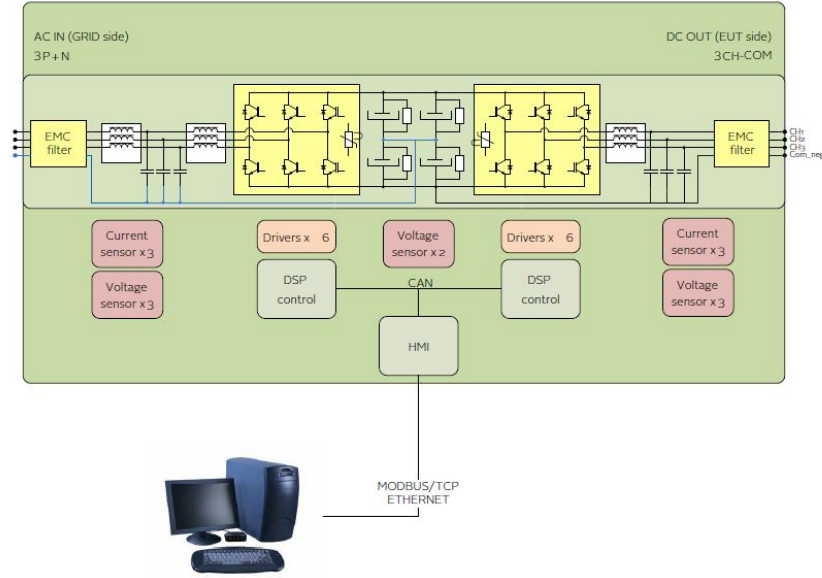


Figure 2.1: Cinergia offers 4 quadrants power conversion solution using the same digital control and Hardware systems [16].

2.2.2 D3 Engineering case

D3 Engineering is a company specialized in outsourcing product development for embedded solutions. They have worked with very important partners like Intel or Texas Instruments for applying digital power solutions.

For example D3 and Intel have created a control platform for bidirectional DC/DC converter in order to accelerate the development of systems with small fast-spinning motors as e-turbos, UAVs, surgical instruments or high-speed pumps.

In collaboration with Texas instruments they have developed a reference design for an automotive engine start-stop boost converter solution. It is capable to deliver steady voltage to vehicle electronics event during voltage drop events such as engine start-up.

Figure 2.2 shows the solution diagram.

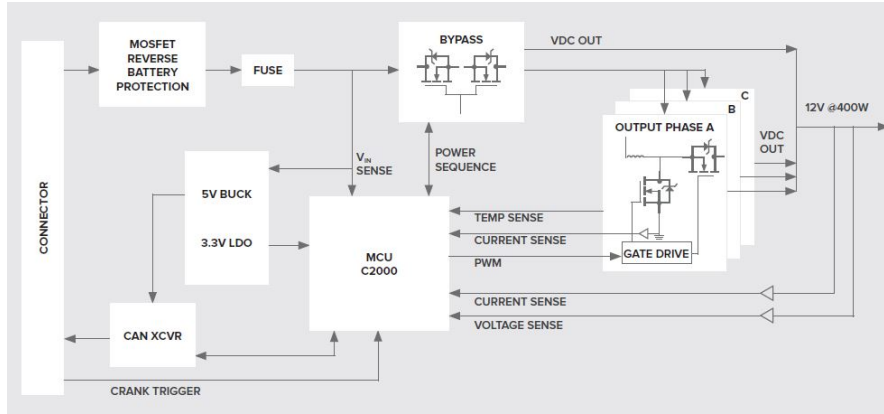


Figure 2.2: D3 Engineering start-stop automotive application boost converter reference design, using digital control systems [17].

2.3 Power converter selected

The implementation of a digital controller can be done in every type of power converter, but for practical and economic aspects the case under study will be a commercial product from PREMIUM S.A. company.

Figure 2.3 shows a picture from the selected power converter.



Figure 2.3: CRS-500 DC/DC push-pull power converter selected for implementing digital control techniques [15].

The specifications of the CRS-500 DC/DC power converter are:

- Manufacturer: PREMIUM S.A.
- Model: CRS-500
- Type: DC/DC isolated converter
- Topology: Push-pull
- Power: 500 [W]
- Input Voltage: 110 [V]
- Output Voltage: 48 [V]
- Actual Controller: Op. Amplifiers + UC2525 from Texas Instruments

Chapter 3

Converter modeling

Switching power converters are electronic devices that manage energy stored by inductive and capacitive components like capacitors and inductors, using switching devices to drive the power from source to load. It means that its electrical circuits are changing continuously from one state to another, introducing non-linearities for voltage and current magnitudes between these states.

These discontinuities are not desired for converter models because non-linear control techniques are sophisticated. It is better to work with classic linear control techniques, nevertheless it demands to have continuous-time expressions for voltage and current.

Averaged steady-state space model is widely used for power converters modelling, it introduces averaged values for the voltages and currents varying between possible different states. This model is introduced by *Fundamentals of Power Electronics* [2].

It is known as steady-state because it linearizes the discontinuities with small-signal model of the variable under study, and also averages its values between different states. It means that the method assumes small ripple approximation and small disturbances from the steady-state. This model can always be obtained if the state equations can be written.

Each reactive component (capacitors and inductors) introduces a state space variable, expressions for first-order differential equations introduced by these components are:

$$\begin{cases} \frac{di_L}{dt} = \dot{i}_L = \frac{1}{L}v_L \\ \frac{dv_C}{dt} = \dot{v}_C = \frac{1}{C}i_C \end{cases}$$

3.1 Push-pull converter schematic and state space equations

Figure 3.1 shows the electric schematic of a push-pull DC/DC power converter. Notice that this topology have two transistors (M1, M2) and two diodes (D1, D2), the combinations from this components introduce the different states.

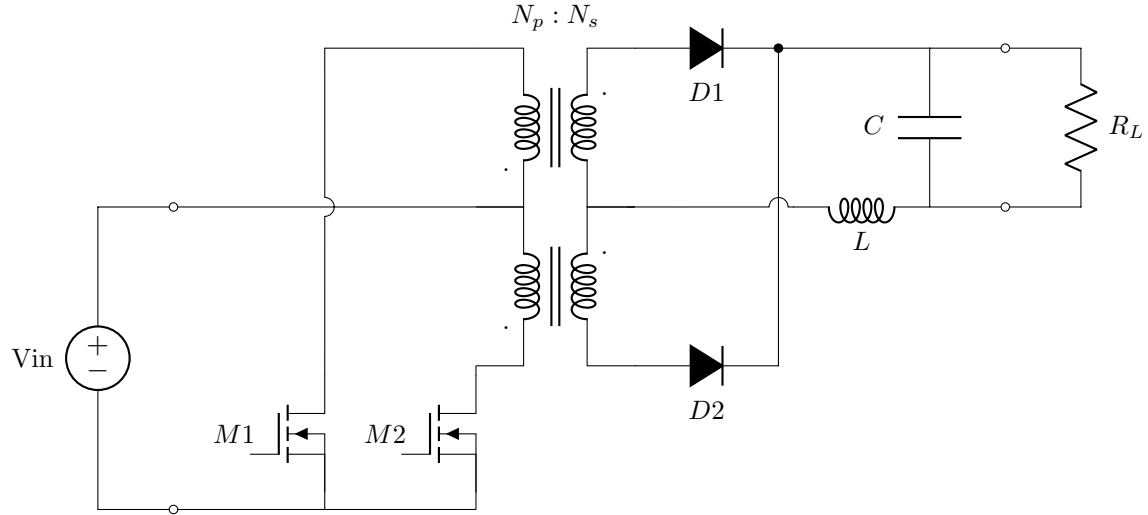


Figure 3.1: Push-pull converter topology schematic.

Maximum number of possible states are defined by the type of switching device (active or passive) and the total number of them.

In this case M1 and M2 can not be active at the same time, this will cause an issue known as shoot-through. For avoid the problem it is required to have a dead-time between transistors conductions, this will be detailed in the implementation section. With this assumption the total number of states is 4, table 3.1 specifies for each possible state if switching components are conducting or not.

Table 3.1: Push-pull converter possible state space summary table.

STATE	M1	M2	D1	D2
S1	ON	OFF	ON	OFF
S2	OFF	OFF	ON	ON
S3	OFF	ON	OFF	ON
S4	OFF	OFF	ON	ON

Taking the schematic with list of multiple states, the electric dispositions will be analyzed to derive the desired expressions and consequently obtain the state space model.

Notice that $S2 = S4$ and expressions from $S4$ will not be analyzed.

State 1 (S1): M1 ON / M2 OFF / D1 ON / D2 OFF

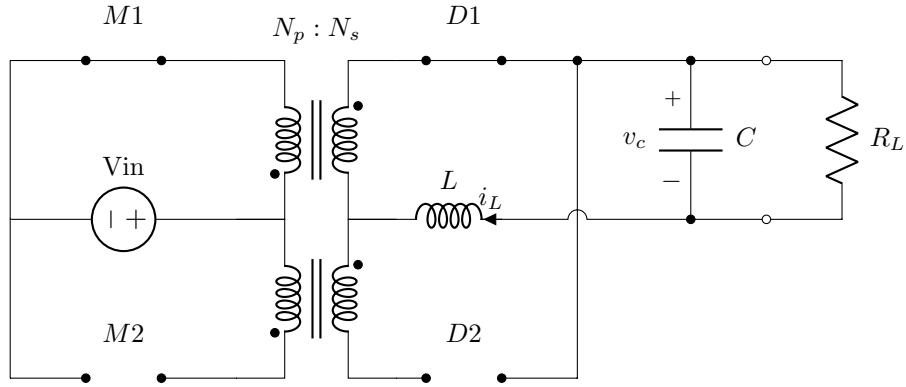


Figure 3.2: Push-pull state 1 electric analysis used to derive state space equations.

The state-space equations in this case are:

$$\begin{cases} \dot{i}_L = \frac{1}{L} \frac{N_2}{N_1} v_{in} - \frac{1}{L} v_c \\ \dot{v}_c = \frac{1}{C} i_L - \frac{1}{CR} v_c \end{cases}$$

State 2 (S2): M1 OFF / M2 OFF / D1 ON / D2 ON

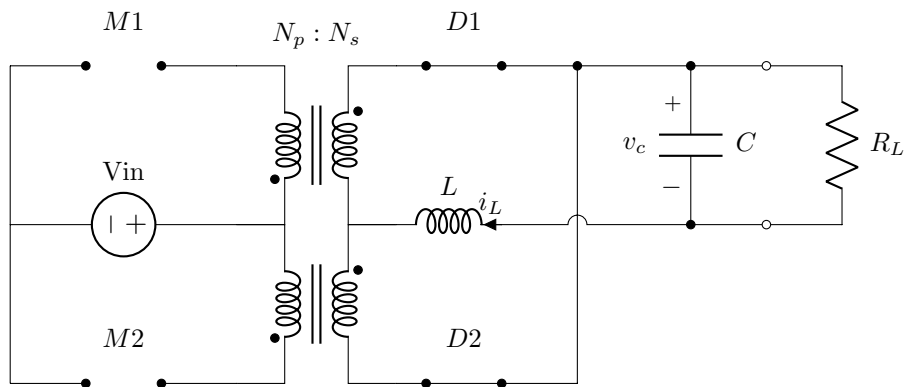


Figure 3.3: Push-pull state 2 electric analysis used to derive state space equations.

The state-space equations in this case are:

$$\begin{cases} \dot{i}_l = -\frac{1}{L}v_c \\ \dot{v}_c = \frac{1}{C}i_l - \frac{1}{CR}v_c \end{cases}$$

State 3 (S3): M1 OFF / M2 ON / D1 OFF / D2 ON

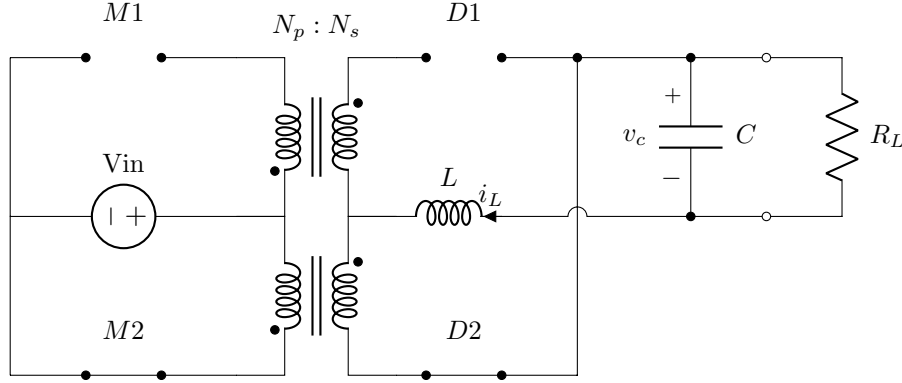


Figure 3.4: Push-pull state 3 electric analysis used to derive state space equations.

The state-space equations in this case are:

$$\begin{cases} \dot{i}_l = \frac{1}{L} \frac{N_2}{N_1} v_{in} - \frac{1}{L} v_c \\ \dot{v}_c = \frac{1}{C} i_l - \frac{1}{CR} v_c \end{cases}$$

Notice that expressions from states S1 and S3 are equal, also for S2 and S4. Which mean that in practice expressions can be defined by two states, consider them S1 and S2.

Non-ideal behaviors from inductive and capacitive components have being neglected because it will be needed to study the printed circuit board characteristics, and the aim of this part is to obtain a model and use it for every push-pull power converter topology, not only for one case of study. Transformer leakage inductance will only affect the transients behaviors and for this reason it is being neglected.

Being t the time domain for $t \in [0, T_s]$ and $u(t)$ the state space variable for $u(t) \in [0, 1]$ the converter model can be defined by two sets of equations, one for S1 and the other for S2 state:

For S1: $u(t) = 1, [0 \leq t \leq T_{on}]$

$$\begin{cases} \dot{i}_l = \frac{1}{L} \frac{N_2}{N_1} v_{in} - \frac{1}{L} v_c \\ \dot{v}_c = \frac{1}{C} i_l - \frac{1}{CR} v_c \end{cases}$$

Expressed in matrix notation it results:

$$\underbrace{\begin{bmatrix} \dot{i}_l \\ \dot{v}_c \end{bmatrix}}_{\dot{X}} = \underbrace{\begin{bmatrix} 0 & \frac{-1}{L} \\ \frac{1}{C} & \frac{-1}{CR} \end{bmatrix}}_{A_1} \underbrace{\begin{bmatrix} i_l \\ v_c \end{bmatrix}}_x + \underbrace{\begin{bmatrix} \frac{1}{L} \frac{N_2}{N_1} \\ 0 \end{bmatrix}}_{b_1} v_{in}$$

For S2: $u(t) = 0, [T_{on} \leq t \leq T_s]$

$$\begin{cases} \dot{i}_l = -\frac{1}{L} v_c \\ \dot{v}_c = \frac{1}{C} i_l - \frac{1}{CR} v_c \end{cases}$$

Again, expressed in matrix notation:

$$\underbrace{\begin{bmatrix} \dot{i}_l \\ \dot{v}_c \end{bmatrix}}_{\dot{X}} = \underbrace{\begin{bmatrix} 0 & \frac{-1}{L} \\ \frac{1}{C} & \frac{-1}{CR} \end{bmatrix}}_{A_2} \underbrace{\begin{bmatrix} i_l \\ v_c \end{bmatrix}}_x + \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{b_2} v_{in}$$

It is needed to include the state variable $u(t)$ in the model expression, this is for having a common equation for every state. For doing so, it is used the bilinear model expression:

$$\dot{X} = (A_1 x + b_1 v_{in})u + (A_2 x + b_2 v_{in})(1 - u)$$

In our case $A_1 = A_2 = A$ and $b_2 = 0$, the resultant expression for model the converter becomes:

$$\begin{aligned} \dot{X} &= Ax(t) + b_1 v_{in}(t)u(t) \implies \\ \begin{bmatrix} \dot{i}_l \\ \dot{v}_c \end{bmatrix} &= \begin{bmatrix} 0 & \frac{-1}{L} \\ \frac{1}{C} & \frac{-1}{CR} \end{bmatrix} \begin{bmatrix} i_l \\ v_c \end{bmatrix} + \begin{bmatrix} u \frac{1}{L} \frac{N_2}{N_1} \\ 0 \end{bmatrix} v_{in} \end{aligned} \quad (3.1)$$

3.2 State space averaged model

In order to avoid discontinuities it is common to take the averaged values from output voltage and inductor current (state space variables), it is also known as state space averaged model. The advantage of doing so is that control variable $u(t)$ is continuous and consequently classic control design techniques can be used. Drawback is that ripples are not modeled and it is needed to assume low ripple approximation in the steady state

of the converter. This Model is introduced by *Fundamentals of Power Electronics* [2], chapter 7.

Averaged model uses the perturbed and the average values as follows:

$$\begin{aligned} u(t) &= D + \hat{d}(t) \\ v_{in}(t) &= V_{in} + \hat{v}_{in}(t) \\ x(t) &= X + \hat{x}(t) \end{aligned}$$

resulting:

$$\begin{aligned} \dot{\hat{X}} &= A\hat{x}(t) + b_1 v_{in}(t) u(t) \\ \dot{\hat{X}} + \hat{X} &= A(X + \hat{x}(t)) + b_1(V_{in} + \hat{v}_{in}(t))(D + \hat{d}(t)) \\ \hat{X} &= A\hat{x}(t) + b_1(D\hat{v}_{in}(t) + \hat{d}(t)V_{in}) \end{aligned}$$

Finally the linearized, small signal, averaged model obtained is:

$$\begin{bmatrix} \hat{i}_l \\ \hat{v}_c \end{bmatrix} = \begin{bmatrix} 0 & \frac{-1}{L} \\ \frac{1}{C} & \frac{-1}{CR} \end{bmatrix} \begin{bmatrix} \hat{i}_l \\ \hat{v}_c \end{bmatrix} + \begin{bmatrix} D\frac{1}{L}\frac{N_2}{N_1} \\ 0 \end{bmatrix} \hat{v}_{in}(t) + \begin{bmatrix} V_{in}\frac{1}{L}\frac{N_2}{N_1} \\ 0 \end{bmatrix} \hat{d}(t) \quad (3.2)$$

3.3 Transfer functions

The following expression is a generalization of the nonlinear continuous averaged model in terms of perturbations and steady-state.

$$\{\tilde{1}s - [DA_1 + (1-D)A_2]\}\hat{X}_A(s) = \overbrace{[(A_1 - A_2)X_A + (b_1 - b_2)V_{in}]}^K \hat{D}(s) + \{Db_1 + (1-D)b_2\}\hat{V}_{in}(s)$$

Taking the same considerations from 3.1 ($A_1 = A_2 = A$ and $b_2 = 0$):

$$\begin{aligned} \{\tilde{1}s - A\}\hat{X}_A(s) &= K\hat{D}(s) + Db_1\hat{V}_{in}(s) \Rightarrow \\ \hat{X}_A(s) &= (\tilde{1}s - A)^{-1}[Db_1\hat{V}_{in}(s) + K\hat{D}(s)] \end{aligned}$$

Where:

$$(\tilde{1}s - A)^{-1} = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} s - \begin{bmatrix} 0 & \frac{-1}{L} \\ \frac{1}{C} & \frac{-1}{CR} \end{bmatrix} \right)^{-1} = \begin{bmatrix} s & \frac{-1}{L} \\ \frac{1}{C} & s + \frac{1}{CR} \end{bmatrix}^{-1} \Rightarrow$$

$$(\tilde{1}s - A)^{-1} = \frac{1}{s(s + \frac{1}{CR}) - (\frac{1}{C} \frac{-1}{L})} \begin{bmatrix} s + \frac{1}{CR} & \frac{1}{L} \\ \frac{-1}{C} & s \end{bmatrix}$$

It results:

$$\begin{bmatrix} \hat{I}_l(s) \\ \hat{V}_c(s) \end{bmatrix} = \frac{1}{s^2 + s\frac{1}{CR} + \frac{1}{CL}} \begin{bmatrix} s + \frac{1}{CR} & \frac{1}{L} \\ \frac{-1}{C} & s \end{bmatrix} \left[\begin{bmatrix} D \frac{N2}{N1} \frac{1}{L} \\ 0 \end{bmatrix} \hat{V}_{in}(s) + \begin{bmatrix} V_{in} \frac{N2}{N1} \frac{1}{L} \\ 0 \end{bmatrix} \hat{D}(s) \right] \quad (3.3)$$

Being:

$$\hat{X}_A(s) = \begin{bmatrix} \hat{I}_l(s) \\ \hat{V}_c(s) \end{bmatrix}$$

From 3.3 can be derived the transfer functions of our model, the most important for designing the control strategy are the output to control $G_d(s)$, and output to input $G_g(s)$. Subindex i_l will define the inductor current while v_c will express capacitor and output voltage. The expressions are defined as:

For $\hat{V}_{in}(s) = 0$

$$G_d(s) = \frac{\hat{X}_a(s)}{\hat{D}(s)} \quad (3.4)$$

For $\hat{D}(s) = 0$

$$G_g(s) = \frac{\hat{X}_a(s)}{\hat{V}_{in}(s)} \quad (3.5)$$

Applying considerations 3.4 and 3.5 to equation 3.3, result the following transfer functions:

$$G_{vcd}(s) = \frac{\hat{V}_{out}(s)}{\hat{D}(s)} = V_{in} \frac{N2}{N1} \frac{1}{(LCs^2 + \frac{L}{R}s + 1)} \quad (3.6)$$

$$G_{vcg}(s) = \frac{\hat{V}_{out}(s)}{\hat{V}_{in}(s)} = D \frac{N2}{N1} \frac{1}{(LCs^2 + \frac{L}{R}s + 1)} \quad (3.7)$$

$$G_{ild}(s) = \frac{\hat{I}_l(s)}{\hat{D}(s)} = V_{in} \frac{N2}{N1} \frac{Cs + \frac{1}{R}}{(LCs^2 + \frac{L}{R}s + 1)} \quad (3.8)$$

$$G_{ig}(s) = \frac{\hat{I}_l(s)}{\hat{V}_{in}(s)} = D \frac{N2}{N1} \frac{Cs + \frac{1}{R}}{(LCs^2 + \frac{L}{R}s + 1)} \quad (3.9)$$

3.4 Steady State Analysis

This section wants to study the steady state performance given for the commercial selected converter and doing so, verify the output LC filter values needed for control designs. Output voltage and inductor current ripples will be derived and then measured experimentally.

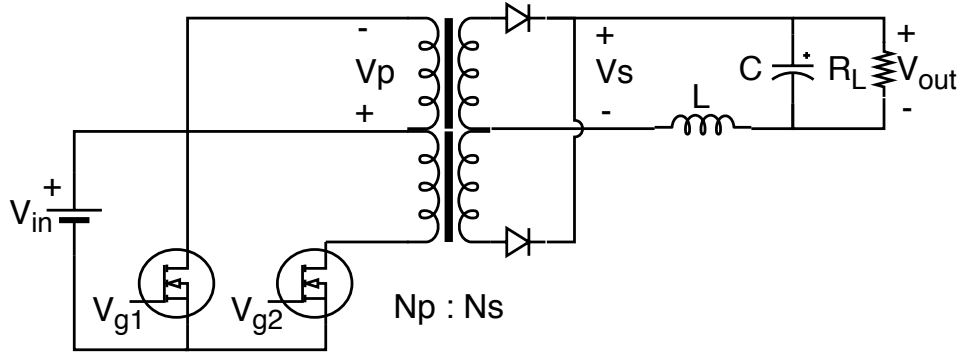


Figure 3.5: Push-pull primary V_p , and secondary V_s side voltage polarity indications.

If we take the push-pull converter output voltage expression and compare it with the buck topology, it is easy to conclude that they are proportional.

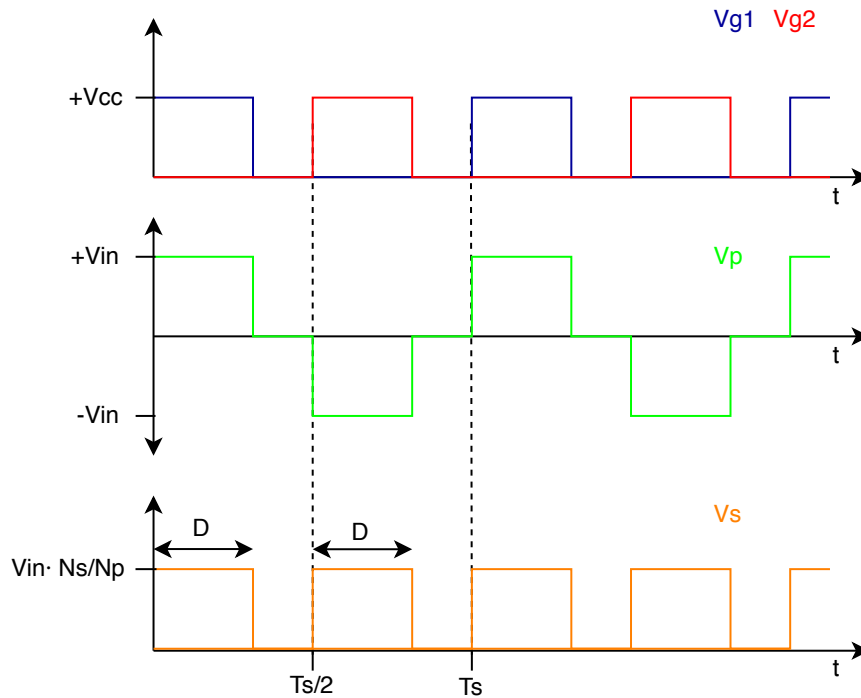


Figure 3.6: Push-pull input and output transformer switching voltage resultant waveforms, used to justify the buck converter analogies.

Figure 3.6 describes push-pull transformer voltages in the primary and secondary side for each period of time, showing that the output voltage of a push-pull converter is a reduced version of the maximum secondary side voltage V_s , which demonstrates that it behaves like a buck converter.

Buck converter output voltage expression is:

$$V_{out} = DV_{in}$$

Push-pull output voltage expression is:

$$V_{out} = 2DV_{in} \frac{N_s}{N_p} = 2DV_s$$

Then, if we take the secondary side diode rectified voltage V_s waveform, the ripple expressions from buck converter can be used for the push-pull topology. All expressions used in this section are derived from the book "Fundamentals of power electronics" [2], sections 2.1, 2.2 and 6.3.

For practical reasons the converter used in this analysis is CRS-120 from the same family of the selected CRS-500, but with 200[W] and 24[V] input voltage instead of 110[V].

Next figures 3.7 and 3.8 show the experimental results obtained for output voltage ripple (ΔV_{out}) and inductor current ripple (ΔI_L). It have been used Tektronix TDS3230 model with a special noise sense probe to take the output voltage and for the current ripple it was used a LEM current probe.

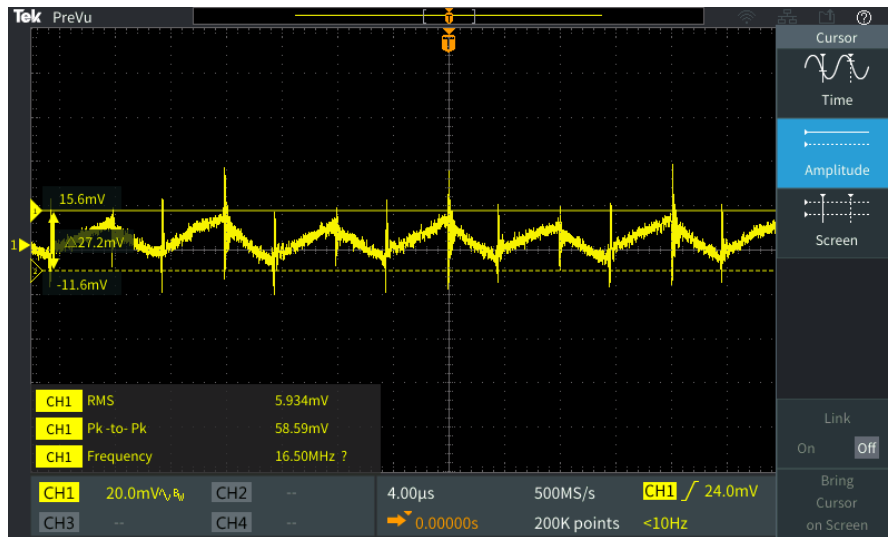


Figure 3.7: Output voltage ripple $\Delta V_{out} = 27.2[mV]$, measured with TDS3230 oscilloscope and ripple probe.



Figure 3.8: Inductor current ripple $\Delta I_L = 76[mA]$, measured with TDS3230 oscilloscope and LEM current probe.

Using the measured current ripple $\Delta I_L = 76[mA]$ and inductor design equation results $L = 2.6[mH]$. In order to check the inductance value it have been measured with an LC meter giving $L = 306[\mu H]$.

Inductance value is closely related with the Inductance Current Ratio (ICR) parameter, it is defined as $ICR = \frac{\Delta I_L}{I_{Lrms}}$. As a design rule $0.05 < ICR < 0.3$ for acceptable current ripple values, and usually $ICR = 0.25$.

Measuring converter $I_{rms} = 2.58[A]$ which using ICR expression means that current ripple should be $\Delta I_L = 0.25 * 2.58 = 0.645[A]$. Taking design expression for minimum inductance give:

$$L = V_{out} \frac{0.5 - D}{f \Delta I_L} = 48 \frac{0.5 - 0.24}{62000 * 0.645} = 312.07[\mu H]$$

The experimental results are not consistent because it is quite difficult to measure inductor current ripple accurately without PCB modifications, despite this the measured value $L = 306[\mu H]$ is consistent with ICR design expression.

For the capacitance it was only needed to read the package, giving $C = 780[\mu F]$. Taking the expression for the minimum output capacitance [18] results:

$$C_{min} = \frac{\Delta I_L}{8f \Delta V_{out}} = \frac{0.645}{8 * 62000 * 0.0272} = 47.8[\mu F]$$

Notice that real value is 16 times higher than the theoretic minimum recommended capacitor, but it is not too strange to oversize the output capacitance value in order to compensate non-ideal effects produced by ESR and ESL.

With CRS-500 converter the same issues were detected, for that reason the output filter check was done using the LC meter for inductance value calculation and again reading the output capacitors labels.

Chapter 4

Voltage Mode Control Design

This chapter includes a set of designs to regulate the selected push-pull topology, each section explain how designs have being performed and raise some solutions to be selected after the simulation chapter.

Power converter selected acts as a voltage source and for that reason it must regulate the output voltage, remaining invariant to different disturbance sources like the load, line and duty cycle perturbations. For this reason it make sense to take the output voltage in the compensation loop for avoid such changes. This technique is widely known as the Voltage Mode Control (VMC).

Figure 4.1 is the block diagram for VMC, the power stage is defined by its transfer function expressions and the output voltage $V_{out}(s)$ is the feedback signal. This sensed voltage passes through a sensing block $H(s)$ and negatively compared to the reference V_{ref} it is processed by the controller $G_c(s)$, producing the compensation signal $V_c(s)$ that actuates on the PWM and finally switching DC/DC converter transistors.

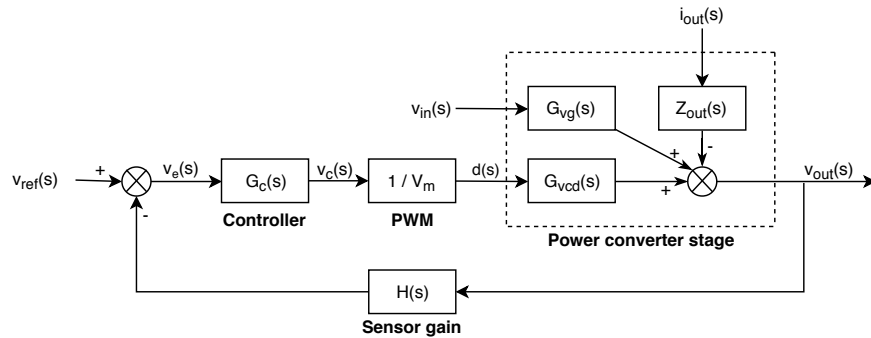


Figure 4.1: Voltage mode control block diagram proposed for push-pull power conversion topology.

To design the converter controller it is needed to define a set of values from the power stage, the most important ones are listed in following table 4.1:

Table 4.1: Selected CRS-500 push-pull relevant converter parameters [15].

Converter parameters	
V_{in}	110 [V]
V_{out}	48 [V]
P_{out}	500 [W]
η	91 %
$N1 : N2$	11:9
L	71.1 [μ H]
C	6000 [μ F]
f_{sw}	124 [kHz]

4.1 Uncompensated closed-loop response

The first step is to study the stability of open-loop gain expression. The system under study will be described as the plant with a transfer function given by expression 3.6, $G_{v_{cd}}(s)$.

Being $\frac{1}{V_m}$ the Pulse-Width Modulator (PWM) equivalent value and G_c the compensator transfer function, the loop can be defined as $T(s)$:

$$T(s) = H(s)G_{v_{cd}}(s)G_c(s)\frac{1}{V_m} \quad (4.1)$$

Notice that in this case $G_c = 1$ for avoid the controller effects and study only the plant stability. Consider the closed loop from previous expression 4.1, $T_{cl}(s)$ as:

$$T_{cl}(s) = \frac{1}{H(s)} \frac{T(s)}{1 + T(s)} \quad (4.2)$$

It can be proved that taking the nominal values for input voltage, transformer turns ratio and nominal duty cycle the converter output voltage obtained is the desired 48[V], as next figure shows:

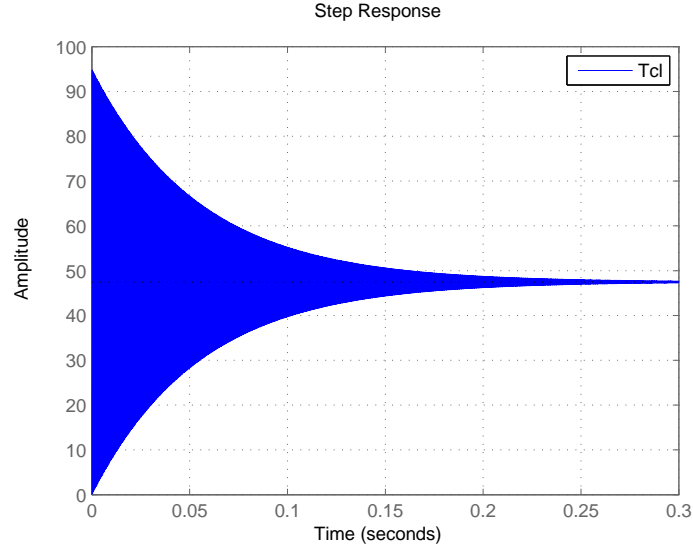


Figure 4.2: Push-pull converter uncompensated closed-loop output voltage response under reference signal step.

From figure 4.2 it can be appreciated that the system has an oscillating under-damped response, which reaches a stable value of 48[V] approximately after 300 [ms]. It is because the phase margin is practically zero, figure 4.3 shows the bode plot obtained from the previous step response and its given phase margin = 0.2° .

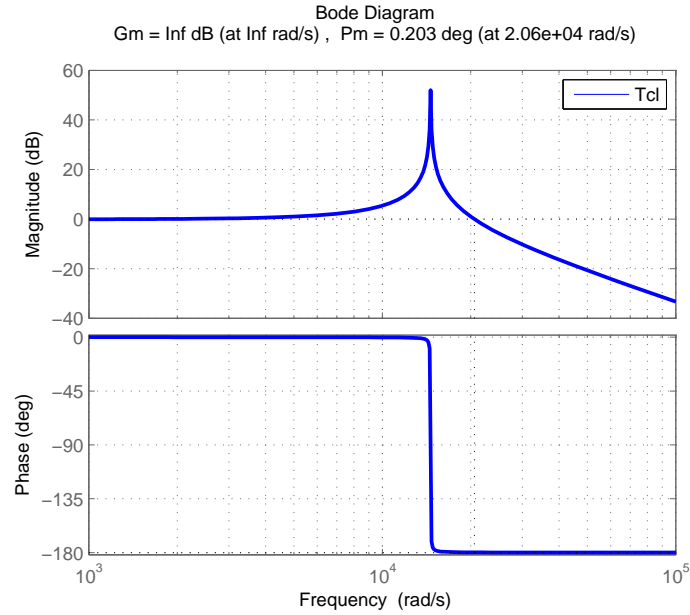


Figure 4.3: Push-pull converter uncompensated closed-loop bode plot shows that phase margin tends to zero ($PM = 0.2^\circ$).

4.2 Proportional Integral compensator

From the obtained uncompensated closed-loop response, a Proportional Integral (PI) controller is proposed, under the following design goals:

1. Null Steady state error
2. Phase Margin = 60°

The PI controller transfer function is:

$$G_{cPI}(s) = \frac{k_P s + k_I}{s} \quad (4.3)$$

k_P and k_I are known as the proportional and integral terms respectively. For calculate its values it is needed to obey the following constrains, where w_c is the crossover frequency in [rad/s] and G_{w_c} the gain in [dB] for the design goals.

$$20 \log k_P = -|G_{w_c}| \quad (4.4)$$

$$k_I = k_P \frac{w_c}{10} \quad (4.5)$$

To set w_c it is needed to obtain the open-loop phase for target $PM_d = 60^\circ$. The asymptotic values of the phase is less than the real one and for that reason it is needed to reduce the desired phase margin using the term ϕ . Using PI compensator $\phi = 10^\circ$, resulting:

$$\varphi_{OL} = -180^\circ + PM_d + \phi = -180^\circ + 60^\circ + 10^\circ = -110^\circ$$

Ones open loop margin φ_{OL} is obtained, using the bode plot of the open-loop response it is possible to know the phase and gain of the tentative crossover frequency. The following figure 4.2 shows the values used for calculating this case:

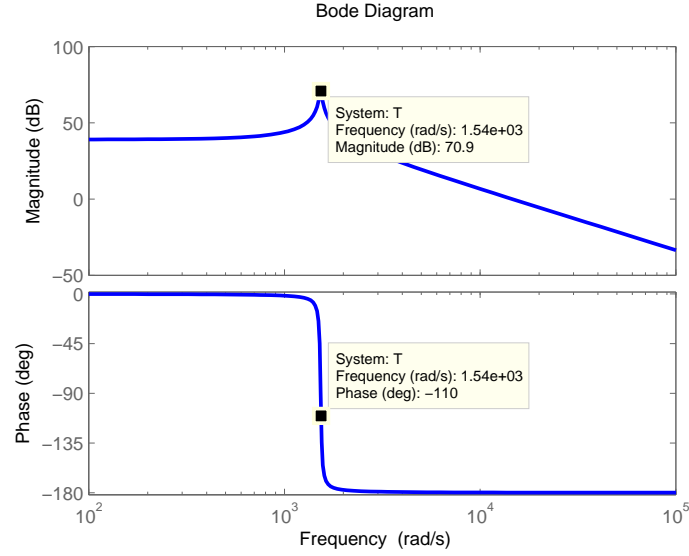


Figure 4.4: PM and w_c obtained from open-loop bode plot for PI controller design.

Notice in 4.4 that for the desired PM, the crossover frequency is $w_c = 1540[rad/s]$, with a gain $|G_{w_c}| = 70.9[dB]$. According to the expressions defined, PI controller results:

$$G_{c_{PI}}(s) = \frac{2.884e^{-4}s + 4.441e^{-2}}{s} \quad (4.6)$$

With this PI compensator design it is obtained the following bode plot and step response:

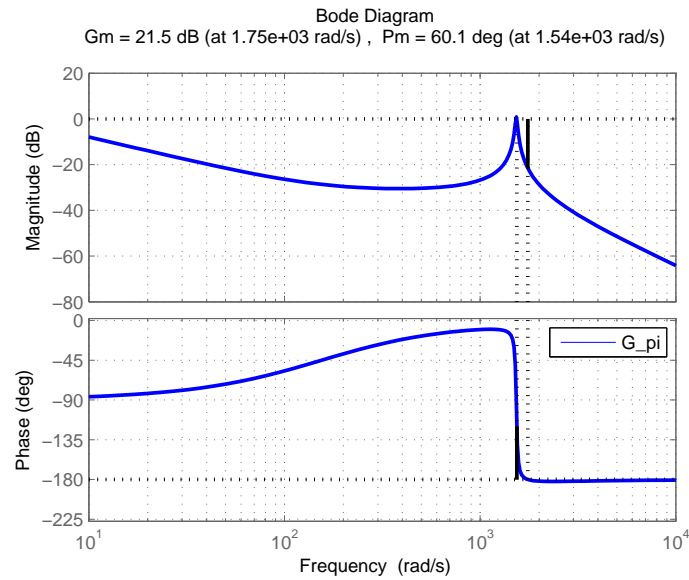


Figure 4.5: PI controller applied to push-pull converter resultant bode plot.

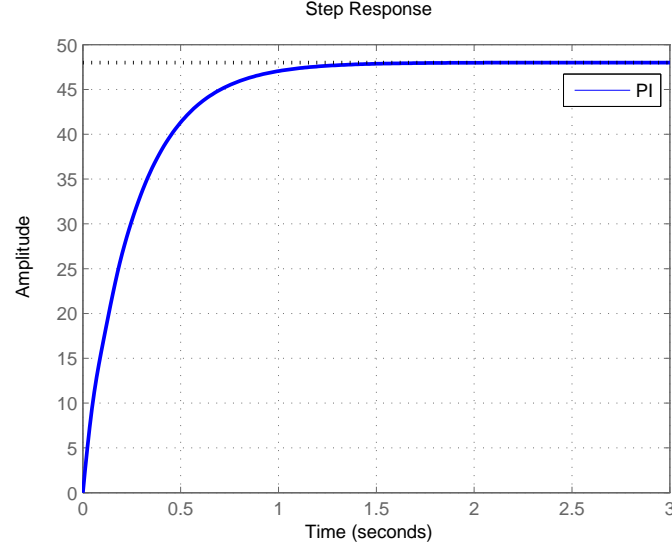


Figure 4.6: PI proposed controller with $PM=60$ closed-loop converter step response results to be too slow.

Notice in the bode plot that $PM \approx 60$ and in the step figure can be appreciated that the time response is approximately 1.5 [s] which is too slow. It should be good to add gain for having faster response, Zero-Pole (ZP) compensator should be studied.

4.3 Lead compensator

A Lead controller, also known as Zero-Pole (ZP) is proposed, under the following design goals:

1. Phase Margin = 60°
2. Crossover frequency: $w_{c2} = w_c/3$

Compensator transfer function is:

$$G_{c_{ZP}}(s) = G_{c_o} \frac{\frac{s}{w_z} + 1}{\frac{s}{w_p} + 1} \quad (4.7)$$

Being:

$$w_p = \frac{w_{c2}}{\sqrt{\frac{1-\sin(PM)}{1+\sin(PM)}}}$$

$$w_z = w_p \frac{1 - \sin(PM)}{1 + \sin(PM)}$$

$$G_{c_o} = 10^{-G_{wc_2} - \frac{20 \log \sqrt{\frac{w_p}{w_z}}}{20}}$$

The aim of introducing the Lead controller is to obtain a faster response, and for that reason the new crossover frequency should be at least 3 times lower, introducing more gain, it means $w_{c_2} = 1540/3 \approx 513[\text{rad/s}]$ and $PM = 60^\circ$. After some calculations, the controller expression becomes:

$$G_{c_{zp}}(s) = 2.465 \frac{1 + \frac{s}{1.170e^4}}{1 + \frac{s}{1.639e^5}} = \frac{2.106e^{-4}s + 2.465}{6.099e^{-6}s + 1} \quad (4.8)$$

Giving the following bode plot and step response:

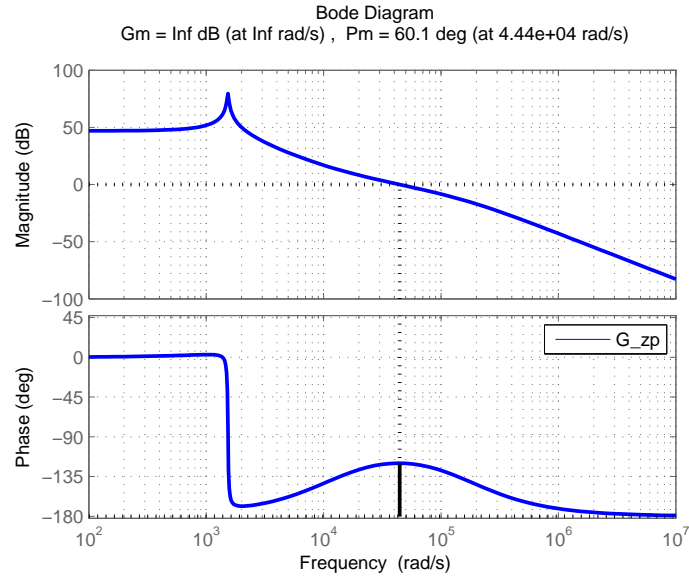


Figure 4.7: Lead compensator with $PM=60$ applied to push-pull converter resultant bode plot.

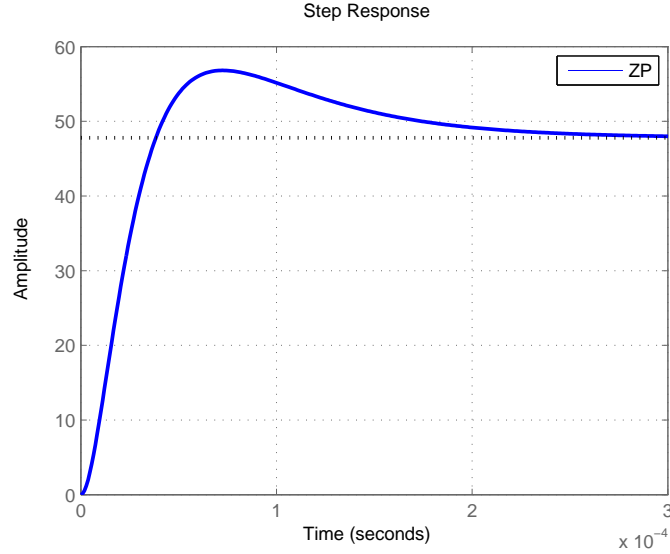


Figure 4.8: Lead controller converter step response is faster than PI but introduce an overshoot.

In this case the response have 30 [ms] transient time, the drawback is that introduces an overshoot of 55[V]. With this option it will be enough to control the converter with good performance.

Depending on the phase margin selected this control law can introduce not-null steady state error, demanding to introduce some DC gain with a PI, but analyzing the results of the steady state value from figure 4.8 it is clearly not the case. Despite that, for experimental purposes it will be studied in the next point, resulting a mix between PI and ZP controller.

4.4 PI+Lead compensator

PI+Lead compensator is the result of mixing the actions of previous PI and Lead controllers. This compensator is also known as 2P2Z because it introduces two poles and two zeros, one of them in the origin from integration part. Compensation design goals are:

1. Null Steady state error
2. Phase Margin = 60°
3. Crossover frequency: $w_{c3} = w_{z_{PI2}} = w_0/10$.

In this case PI compensator part will be modified for having a crossover frequency 10 times higher than the resonance w_0 , while the Lead part will remain exactly as it was

defined before.

PI+Lead controller transfer function is:

$$G_{c_{PI+ZP}}(s) = G_{c_o} \frac{1 + \frac{s}{w_z}}{1 + \frac{s}{w_p}} w_{z_{PI2}} \frac{1 + \frac{s}{w_{z_{PI2}}}}{s} \quad (4.9)$$

Computing $w_{z_{PI2}} = \frac{1/\sqrt{LC}}{10} = 1530/10 = 153.1[rad/s]$, giving the resultant controller expression:

$$G_{c_{PI+ZP}}(s) = \frac{2.106e^{-4}s^2 + 2.498s + 377.4}{6.099e^{-6}s^2 + s} \quad (4.10)$$

With this proposed design can be obtained the following bode plot and step response:

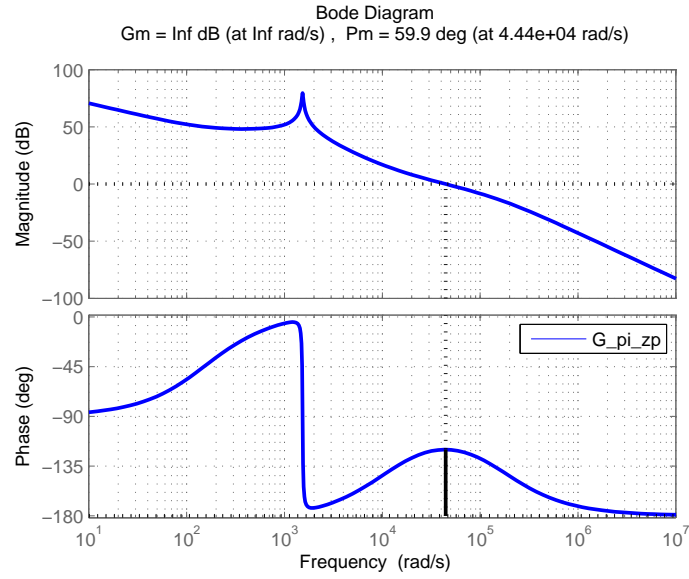


Figure 4.9: PI+Lead controller bode plot shows that system has more DC gain compared with Lead compensator.

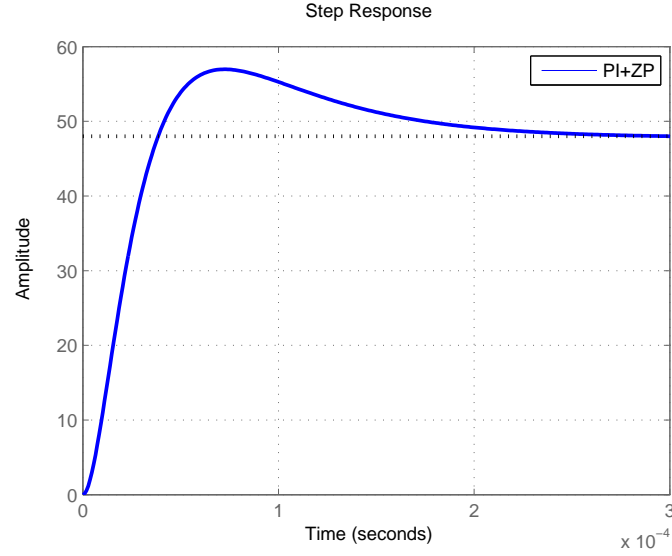


Figure 4.10: PI+Lead controller system step response.

The difference between Lead and PI+Lead is that PI+Lead has more DC gain for lower frequencies due to PI term, and as it was expected the system response remains equal than Lead controller from figure 4.8, same transient time, overshoot and steady state error.

Figure 4.11 compares the effect of phase margin. Notice that the higher PM is, lower overshoots are obtained. Worst case is $PM = 60^\circ$, where the overshoot is around 15%, this value is acceptable because the converter maximum output voltage is into this range.

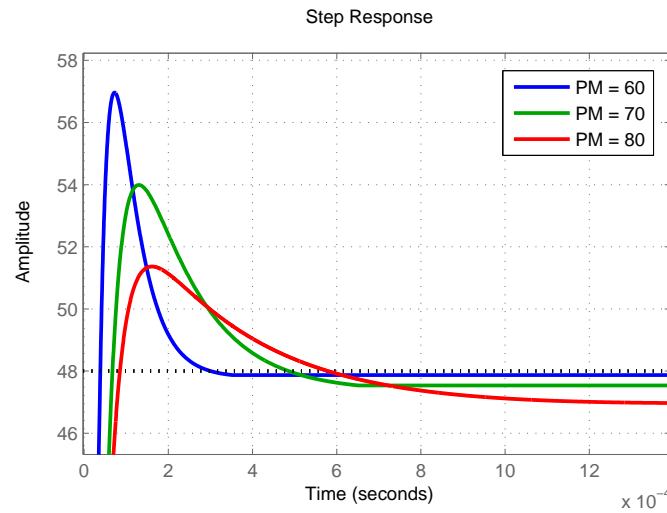


Figure 4.11: PI+Lead compensator push-pull converter step response comparing phase margin effect.

4.5 Proportional Integral Derivative compensator

In order to have several controllers designs, a Proportional Integral Derivative compensator has been studied, parallel implementation is the sum of three parts; Proportional + Derivative + Integral as figure 4.12 shows.

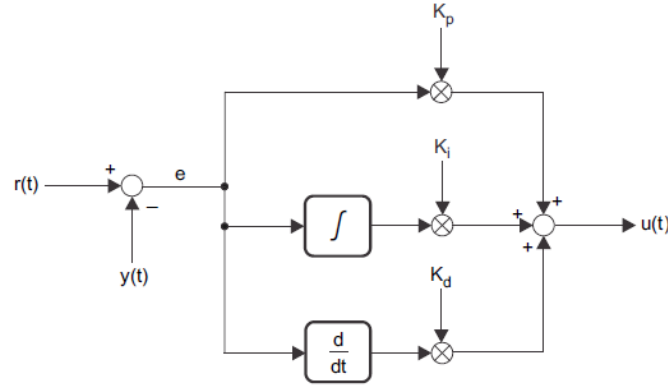


Figure 4.12: PID controller parallel implementation block diagram [13].

The following expression is the parallel implementation in the s-domain of a PID compensator:

$$G_{cPID}(s) = K_p + \frac{K_i}{s} + K_d s \quad (4.11)$$

The design of a PID controller can be done using the PIDtuner tool from Matlab® very easily. It has two different possibilities, design in the time domain by introducing the transient time and behavior or alternatively in the frequency domain, setting the bandwidth and phase margin desired.

Table 4.2 shows multiple PID compensator solutions obtained.

Table 4.2: PID parameters obtained for different phase margin.

PM	K_p	K_i	K_d
60°	0.042	3.11	$1.61e^{-5}$
70°	0.077	9.00	$2.99e^{-5}$
80°	0.102	17.80	$5.10e^{-5}$

The step response is faster for higher phase margin values while overshoot is less than 10% for each case, like the following figure 4.13 shows:

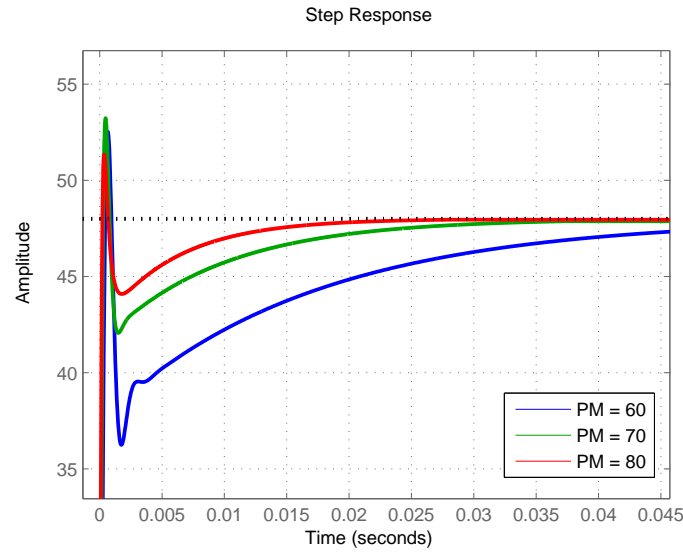


Figure 4.13: PID controller output voltage step response comparing several phase margins.

In order to compare the performance between PI+ZP and PID compensation it was selected 60° phase margin option. Resultant expression is:

$$G_{cPID}(s) = 0.042 + \frac{3.11}{s} + 1.61e^{-5}s \quad (4.12)$$

4.6 PID vs PI+Lead

To select the compensator it is needed to perform more simulations which will be included in the following chapters but as a first approximation next figures show the most relevant differences between two proposed controllers.

In order to compare both controllers, selected phase margin is 70° because for 60° case differences between them are less evident.

Steady state error is zero, the results from the figure show errors due to stabilization time is not reached.

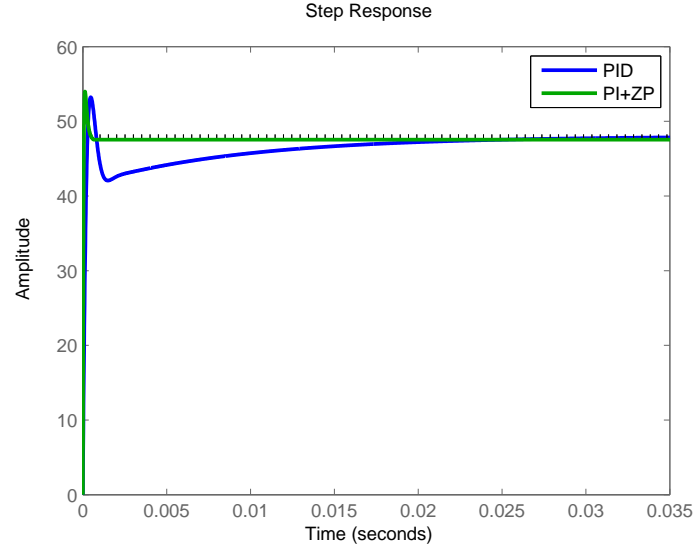


Figure 4.14: Push-pull converter output voltage step response comparing PID vs PI+Lead controllers with 70° phase margin.

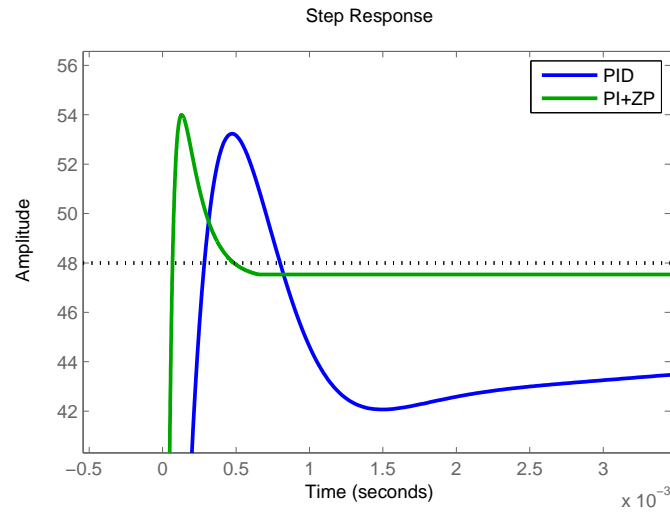


Figure 4.15: PI+Lead is significantly faster than PID as this step response detail show.

The main difference between proposed solutions is the transient time, while overshoot is practically the same, see the differences summarized in next table 4.3.

Table 4.3: Performance comparison between PID and PI+Lead controllers.

Type	Step response	Overshoot
PID	30 [ms]	12.5%
PI+ZP	0.6 [ms]	10.8%

At this point, PI+ZP seems to be the best solution for transient response because it is by far faster than PID controller and has practically the same overshoot. For selecting the best solution, next chapter introduces more simulations with Simulink and Psim software.

Chapter 5

Simulations

To study the converter behavior under the influence of designed controllers, this section will include simulations using programs like Matlab-Simulink[®] and PSIM[®]. The following sections will be divided in parts where the simulation scenario will progressively closer to the final implementation conditions.

5.1 Converter model

For being more consistent with the derived state space averaged model from equation 3.2, the simulations will be performed with functional blocks instead of an s-domain transfer function expression. Figure 5.1 shows the block diagram used in Simulink.

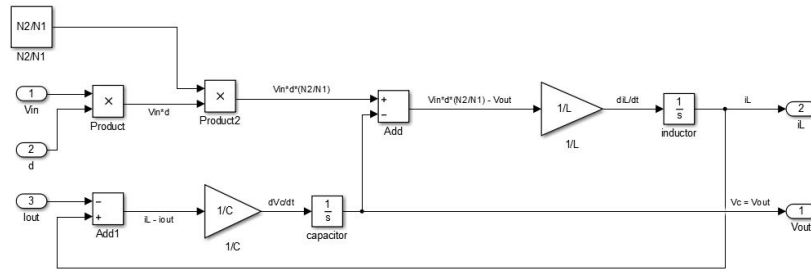


Figure 5.1: Push-pull converter state space averaged model block used in Simulink[®].

First of all, it is needed to check that the state space model used for the following simulations (figure 5.1) matches with the one obtained by mathematical expression, expressed by converter transfer functions from previous sections.

Comparing step responses, the waveform obtained by transfer function has a higher frequency oscillation than block model provably for the Simulink calculation engine, but

the damping response is practically the same. Figure 5.2 demonstrates that converter model is consistent.

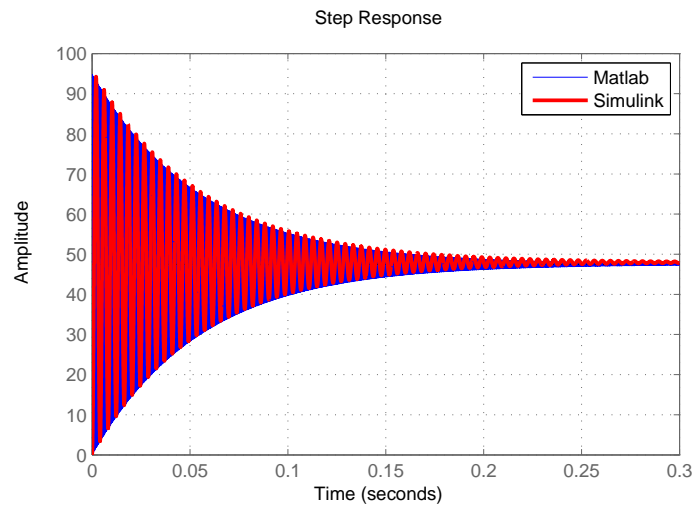


Figure 5.2: Comparing state space model used in Simulink with Matlab transfer function $G_{vcd}(s)$ expressions used during the controllers designs.

5.2 Continuous-time domain

All compensation algorithms have been designed using frequency domain concepts like phase margin and bandwidth, which means continuous time domain. It is known that the final solution will be in discrete-time domain, but the purpose of this section is to study the controllers performance, assuming an analogical implementation.

Figure 5.3 show the voltage mode loop implementation of an analog compensator for the push-pull converter topology selected.

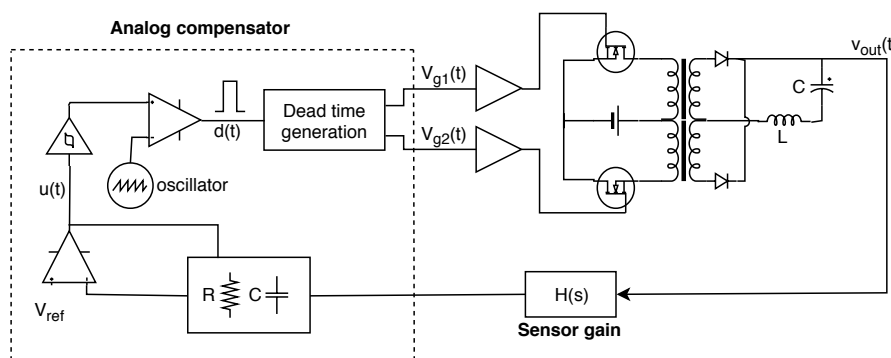


Figure 5.3: Analog voltage mode control block diagram.

Notice that push-pull converter block defined in figure 5.1 PWM block do not implement frequency modulation, control signal $V_{control} = u(t)$ is the mean duty cycle "d" value, being this the input feedback signal.

For the following simulation results, it have been selected the PI+Lead controller case and despite that it is not defined yet, the selected phase margin = 60° .

Figure 5.4 shows the block diagram model used for performing the different perturbation rejection simulations in continuous-time domain.

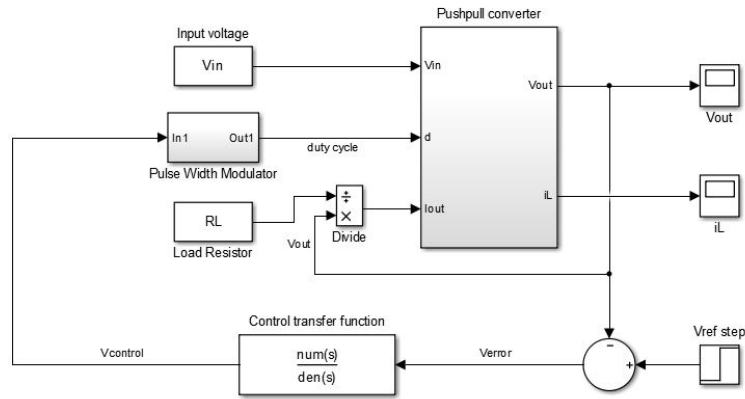


Figure 5.4: Converter closed loop block model used for continuous-time simulations.

Converter dynamic performance is being tested by applying some disturbances in the load (I_{out}), input voltage (V_{in}) and reference signal (V_{ref}) and observing the output voltage signal response.

This simulations are done for knowing the controller transient response, for output load abrupt changes it is known as the load regulation or load disturbance specification, while for input power supply it is known as the line regulation.

5.2.1 Load disturbance

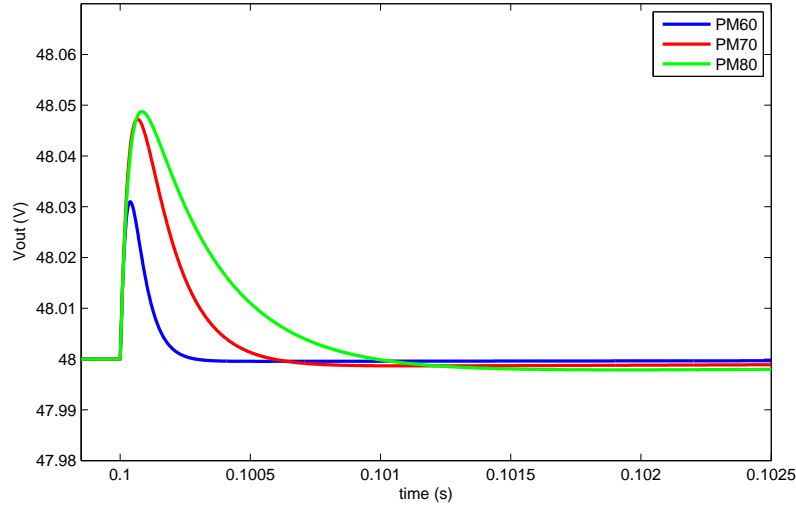


Figure 5.5: Converter output voltage with PI+Lead controller load step response from $P_{out} = 500[W]$ down to $P_{out} = 28.8[W]$.

Figure 5.5 shows the transient response of the converter output voltage for an output current step, from nominal value of 10.4[A] down to 0.6[A] using PI+Lead compensation techniques under different phase margins.

Transient time is the same as the obtained in design section, figure 4.11 but the overshoot is 3 orders of magnitudes less, possibly due to Matlab calculation engine.

5.2.2 Line disturbance

For input voltage disturbances it is done a step-up over the worst case values, from $V_{in_{min}} = 80[V]$ up to $V_{in_{max}} = 130[V]$, being this the minimum and maximum allowed input voltages respectively.

The output voltage waveform resultant is shown in figure 5.6:

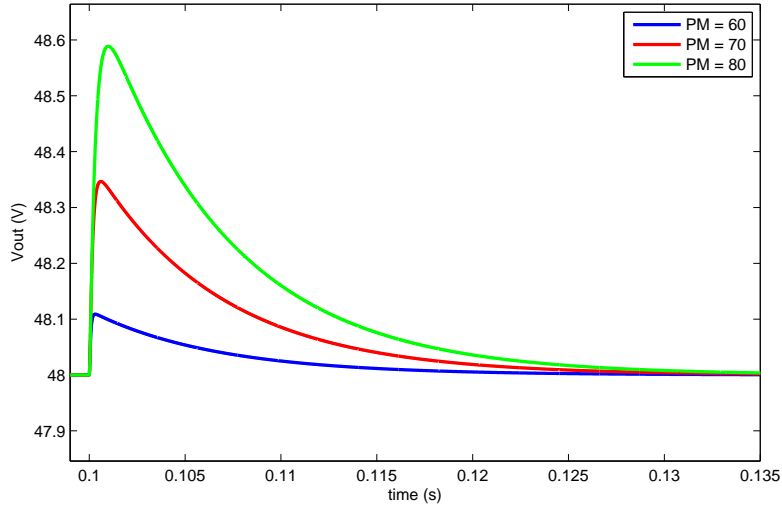


Figure 5.6: Converter output voltage with PI+Lead controller line disturbance rejection, input voltage step response from $V_{in_{min}} = 80[V]$ up to $V_{in_{max}} = 130[V]$.

Notice that the lowest phase margin case, results less overshoot peak and transient response time than other cases. Giving from PM=60° to PM=80° respectively, peak values of 0.21%, 0.73% and 1.25% and transient times of 10[ms], 25[ms] and 35[ms].

5.2.3 Reference tracking

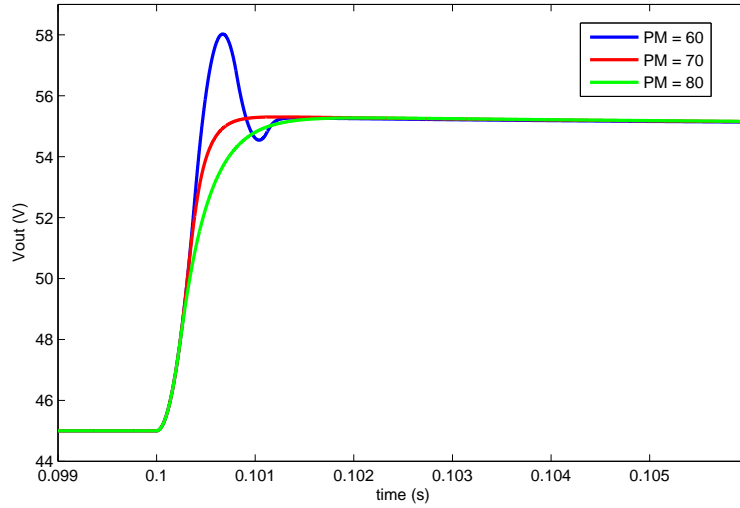


Figure 5.7: Converter output voltage with PI+Lead compensator, reference signal step response from $V_{ref} = 45[V]$ up to $V_{ref} = 55[V]$.

To check the effect of a change in the reference signal during the steady state it is applied a step-up from 45[V] up to 55[V]. The response for $PM = 60^\circ$ is slightly overdamped while the other cases do not have any peak, the drawback is the response is slower.

The most important rejections to disturbances are from input voltage and specially output load because in this application it is not usual to change the reference signal while the converter is working, the power conversion stage is usually used for steady state conditions of reference voltage.

5.2.4 Pulse width modulation effects

Pulse width modulation technique is implemented by comparison of a sawtooth signal with the compensation signal, resulting a train of pulses with a duty cycle proportional to it.

In last simulations PWM block was considered as the mean value V_m of the frequency modulation, resulting an integer value without restrictions. In order to get more realistic results, two effects are being considered:

- State variable limit values $d \in [0, 1]$.
- Switching frequency modulation.

In previous sections the control signal had no restrictions, being freely applied to the model input duty cycle but it is not possible because the values must be limited between the duty cycle capacity, where duty $d \in [0, 1]$.

After some load and line disturbances simulations, the results were exactly the same than without considering duty cycle restriction because the compensation signal never reach the limits.

Figure 5.8 shows the compensator output signal for different PM designs during the load step described in previous load disturbance section (from 500[W] down to 28.8[W]). It is known as control effort and it shows that with PWM restrictions added the result are equal because it never reaches 0 or 1 limits.

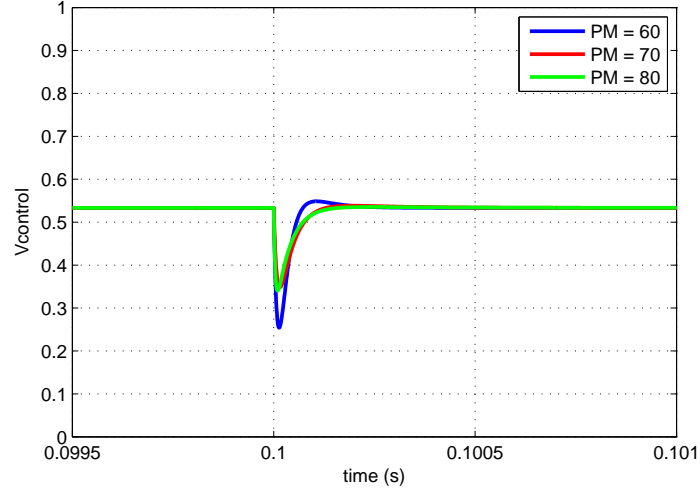


Figure 5.8: PI+Lead Controller output signal load step response from $P_{out} = 500[W]$ down to $P_{out} = 28.8[W]$ never reach saturation, then adding PWM duty cycle limits $[0,1]$ do not affect simulation results.

In the real case duty cycle is not an integer value, pulse width modulation introduces ripples for inductor current and also capacitor voltage. Notice that it introduces two different frequencies, the higher ones are for the switching frequency while other oscillations come from the compensation signal slight variations. Low frequency ripple in output voltage is negligible, take special attention with the vertical scale resolution.

This results do not take in care non-ideal effects from inductive and capacitive component, which by sure will increase the ripple values. This only shows the effect on the control signal due to pulse-width modulation.

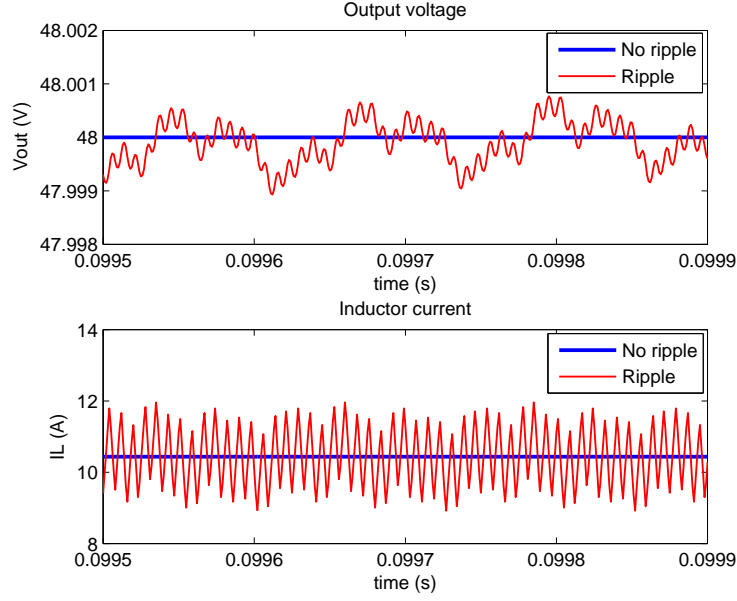


Figure 5.9: Ripple under pulse width modulation effects. Top: Output voltage ripple introduced by PWM block, notice vertical scale values. Bottom: Inductor current ripple.

Figure 5.10 show the response under already described load step comparing the case with PWM frequency modulation versus the mean value approximation.

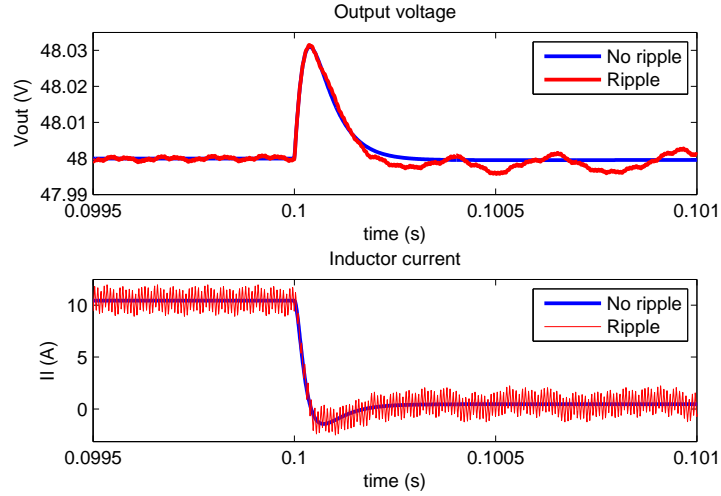


Figure 5.10: PI+Lead with PWM block load step response from $P_{out} = 500[W]$ down to $P_{out} = 28.8[W]$. Top: output voltage. Bottom: inductor current.

Analyzing the control effort with rippled values, again the signal never reaches saturation and consequently the system response remain invariant.

At this point the design is validated by analog compensator implementation, despite this, the objective of this project is to implement digital control techniques in a push-

pull converter and for that reason it is needed to study how discrete-time implementation will affect proposed controller solutions.

5.3 Discrete-time domain

This section wants to show that from an analog domain design, it is possible to implement a controller using algorithms in discrete-time that behaves like in continuous-time. Obviously it have some limitations which will be explored in the following parts to guarantee final implementation viability.

The basic change from analog control is the compensator itself, which now is digital. It implies that input and output from the compensation algorithm must be also digital, and as a result, it demands changes from analog to digital domain because plant stills behave in continuous-time.

Figure 5.11 shows the diagram for a discrete-time implementation of voltage mode control:

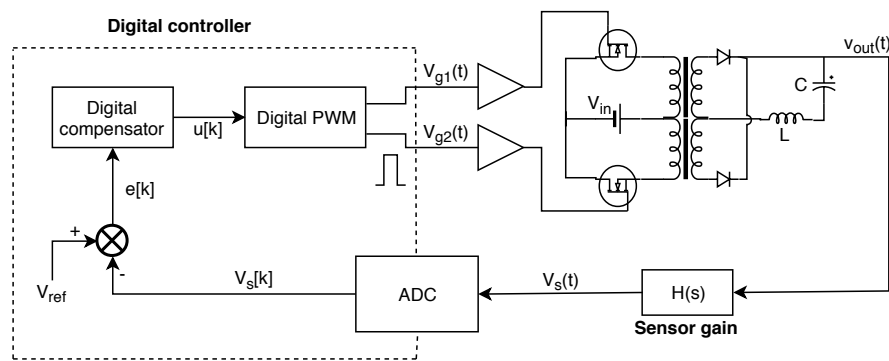


Figure 5.11: Digital compensator voltage mode control block diagram.

Notice that from the analog $V_{out}(t)$ the controller receive a periodically sampled version of this magnitude. Lets consider it as $V_s(k) = V_{out}(k)$ being $k = t_k$ the sampling instants under the sample period T_s .

The sampling of each $V_{out}[k]$ is done by an analog-to-digital converter (ADC) introduced in the next section.

5.3.1 ADC

Analog to digital converter acquires continuous-time signals, takes a while for quantizing them and sending its data values to digital compensation block. In the case of voltage mode control the input values is $V_{out}(t)$ and the output value is an array of samples, let us consider them $V_{out}[k]$ for each sampling instants $k = t_k$.

ADC simulation model

In order to check ADC's behavior, it is being modeled by a switch that is activated each sampling time followed by the quantizer delay block terminated with a sample and hold block. Time delay introduced by quantization is being neglected because it is assumed to be in the range of nanoseconds while sampling period will be in the range of microseconds.

During one sampling time, the value remains equal until next sampling and quantification is done. Next figure shows the model used:

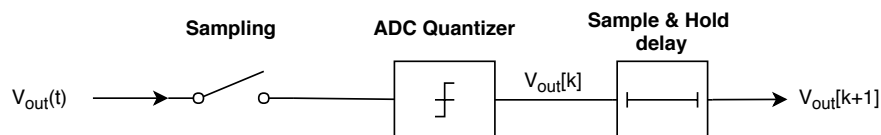


Figure 5.12: Analog to Digital converter model block diagram used for discrete-time simulations.

Sampling time selection

The synchronization of ADC and PWM blocks is crucial in digital power converters, which mean that selecting the sampling rate and switching frequency is one of the most important things to do before starting to design the compensator. Following arguments explain why the selected sampling time will be equal to sampling period.

Most relevant signals; capacitor voltage and inductor current always have high-frequency and baseband (DC) spectrum components. The desired value is always the DC component, but this signals always contain harmonics from switching frequency aliased in the sampled signal regardless of the sampling period time.

Consider figure 5.9 output voltage as the signal to be measured. The desired value is the blue one, while the red is the real waveform.

Taking sampling rates higher than switching frequency will be possible to acquire the oscillations from voltage ripple and also from PWM aliasing, but this is not what it is desired. Another drawback is that for fast sampling periods the available computing

time is being reduced and consequently the digital compensator needs higher clock frequency, which is possible but with some limitations.

In the other hand, taking sampling rates below the switching frequency can introduce non-real oscillations due to the ripples.

Taking sampling rates equal than switching frequency can introduce some offset errors in the measures, in the case of high ripple waveforms the errors can be considerably high, nevertheless the model used is only valid for small ripple approximation and in case of deviations they can be compensated by software measurement calibration. Figure 5.13 from reference [1] shows the effect of sampling at switching frequency offset.

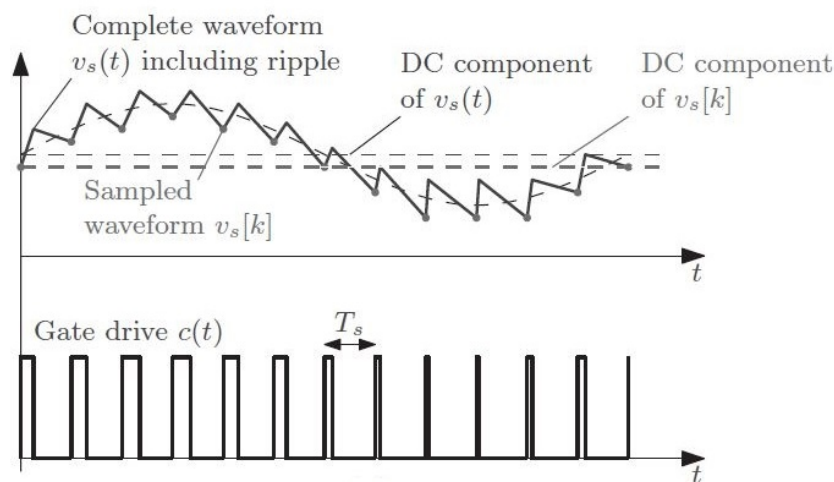


Figure 5.13: Voltage sampling rate equal to switching frequency resultant measurements $v_s(t)$ from reference [1].

For that reasons it is common to use a sampling rate equal to switching frequency, then:

$$T_{sampling} = T_{sw} = \frac{1}{f_{sw}} = \frac{1}{124[kHz]} = 8.0645e^{-06} \approx 8[\mu s]$$

5.3.2 Digital PWM

Digital Pulse-Width Modulation (DPWM) takes the resultant value $u(k)$ from digital compensator in the instant $t = k$, quantizes the value and generates the square signal proportional to control variable at $t = k + 1$ instant. This sequence is repeated continuously for each switching time period where the signal generated by DPWM is known as the duty cycle $d(k)$.

$$d(k) = \frac{u(k)}{N_{bits}}$$

The difference from analog PWM is that resolution is limited by its quantization performance q_D and also the system clock period time T_{clk} .

Generation of resultant duty cycle is limited by compensator clock frequency latency T_{clk} because the minimum resolution time is one line of code and it lacks one time latency. This property is known as the time resolution $\Delta t_{DPWM} = T_{clk}$. Quantization q_D is directly related with the bit-resolution of DPWM block, being:

$$q_D = \frac{T_{clk}}{T_{sw}} = \frac{1}{N_{bits}}$$

In final implementation DPWM block will have enough performance to neglect this issues. For example, with a DSP clock frequency $f_{SYSCLKOUT} = f_{clk} = 100[MHz]$ and a converter switching frequency of $f_{PWM} = 124[kHz]$ we have:

$$PWM_{resolution}(\%) = \frac{f_{PWM}}{f_{SYSCLKOUT}} = 0.124\%$$

DPWM simulation model

DPWM block is implemented by a Digital-to-Analog converter which can be modeled by a sample and hold.

Quantizer delays from ADC and DAC are not included in model simulations. This is due to the sampling time selected, the computation time is $8[\mu s]$ while quantizer times are $75[ns]$ for ADC and $25[ns]$ for DAC, which represents the 0.93% and 0.31% respectively from the period time and then it is being neglected. All the values are obtained from reference [14].

5.3.3 Digital Compensator

From the subtraction of sensed ADC waveform and the reference value is obtained the error signal $e[k]$. Digital compensator computes this value and delivers the discrete-time control signal to DPWM after a time delay t_{calc} .

The mathematical operands of the controller are obtained from the domain transformation of the transfer functions designed. In the following section it is described how can be obtained.

Discretization procedure

There are several techniques used to approximate a continuous-time expression by a discrete-time one, but in essence it is needed to calculate the integral of a function dur-

ing the sampling time. Lets consider the following random function $f(x)$.

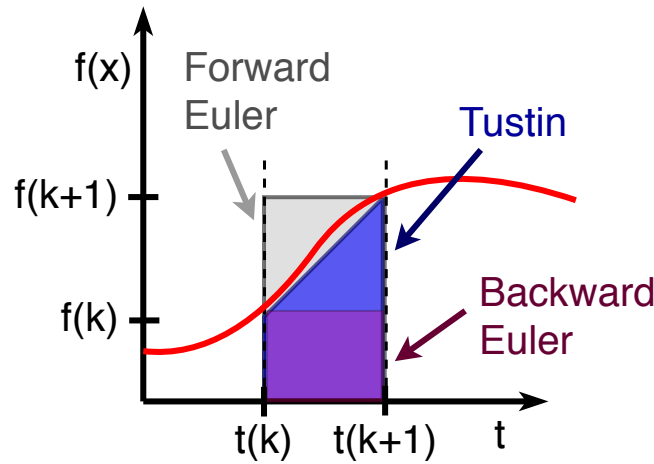


Figure 5.14: Discretization for continuous-time signals can be achieved from different approximation algorithms, Tustin method is commonly used because it shapes better the signals.

Figure 5.14 shows some approximation techniques like the Backward Euler, Forward Euler or Trapezoidal rule also known as Tustin approach. It can be appreciated that the most accurate technique is the Tustin approach and for that reason it is the selected method for the discretized controller transfer function.

Tustin approximation over the sampling time is defined as:

$$\int_{(k-1)T_s}^{(k)T_s} x(\tau) d\tau \approx \frac{T_s}{2} (x(kT_s) + x((k-1)T_s)) \quad (5.1)$$

Which means that over one sampling period the value of the discrete expression is a trapezoidal approximation. In other words, to get the s-to-z domain transformation, the change variable for Tustin approach is:

$$s \longrightarrow \frac{2}{T_s} \frac{1 - z^{-1}}{1 + z^{-1}}$$

Tustin transformation is done using Matlab scripts and resources for saving the procedure for future designs. The code takes designed s-domain transfer function and applies z transform.

Discretized controllers

For example, taking 2P2Z compensator equation 4.10 and sampling period $T_s \approx 8[\mu s]$ under Tustin approach it results the following z domain expression 5.2:

$$G_{c_{PI+ZP}}(s) = \frac{2.106e^{-4}s^2 + 2.498s + 377.4}{6.099e^{-6}s^2 + s} \Rightarrow$$

$$G_{c_{d_{PI+ZP}}}(z) = \frac{22.02z^2 - 49.09z + 20.07}{z^2 - 1.22z + 0.22} \quad (5.2)$$

Notice that the values have only two decimal digits but in real implementation and simulations it have been taken as many digits as it was possible for having more accurate results.

In order to know if the discretized compensator shapes with the analog expression it is compared the voltage reference step response in closed loop.

Doing a 48V reference voltage step, it is being compared the system closed loop response from the continuous-time controller $G_c(s)$ and its discretized version $G_{c_d}(z)$. The resultant dynamic response it is practically the same as figure 5.15 shows. Notice that for digital compensator the signal is constant during each time period.

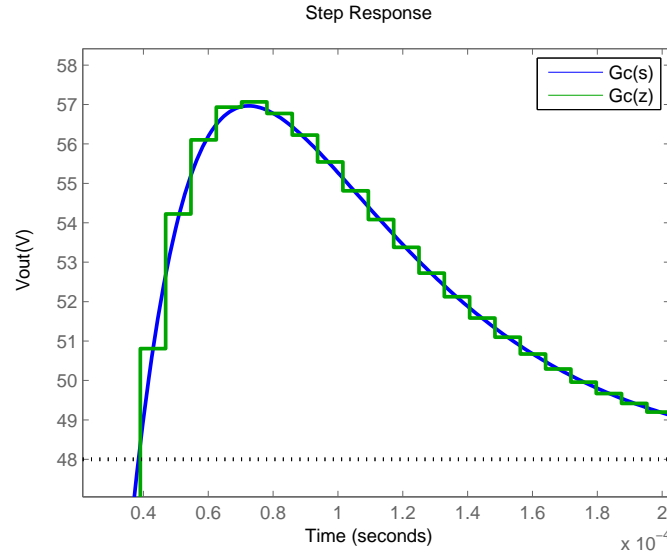


Figure 5.15: Converter closed loop reference step response comparing analog controller $G_c(s)$ and its discretized digital version $G_{c_d}(z)$ for sampling period selected $T_s \approx 8[\mu s]$.

Taking PID compensator equation 4.11 and same sampling period results the following

z-domain expression 5.3:

$$G_{cPID}(s) = K_p + \frac{K_i}{s} + K_d s$$

$$G_{cPID}(z) = K_p + K_i \frac{z T_s}{z - 1} + K_d \frac{z - 1}{T_s} \quad (5.3)$$

5.3.4 Simulink results

In this section are detailed all simulations for digital control solution, in order to validate if the analog designed controllers discretized are good enough approximations to implement the control law. It will be done in a progressive way as it was already commented.

This simulations include the aspects described in ADC and DPWM sections and also includes a pure delay of one cycle from the digital compensator since the duty cycle actualization is done at the end of the sampling period. For example, the value from error at instant $k = 1$ has being processed and compensated in the next sampling time duty cycle actualization, at time $k = 2$.

First simulation approach has being done by Simulink software, and like in continuous-time, system perturbations (load, line and reference signal) are being tested through several suddenly changes and compared its performance with analog designs. Figure 5.16 shows the resultant model used:

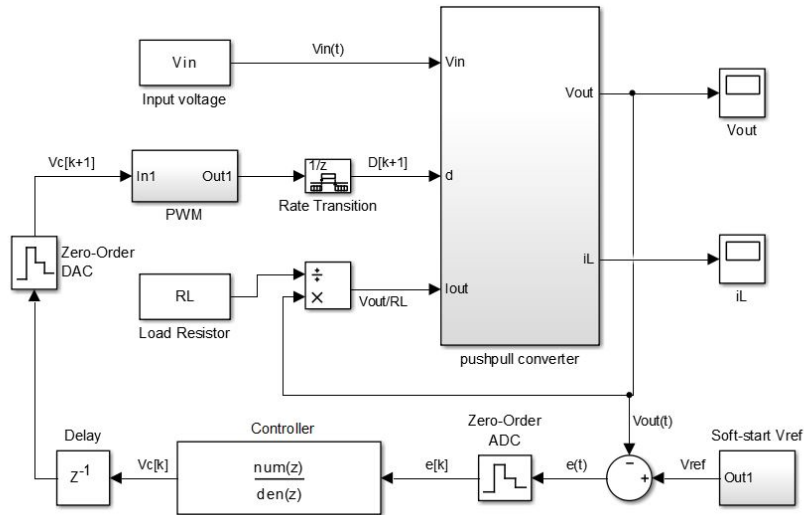


Figure 5.16: Push-pull converter digital control block diagram used for discrete-time simulations done in Simulink.

Notice that the reference signal has a soft-start block because during the input voltage connection start-up the compensation loop became saturated and started to oscillate for such abrupt change in the reference. This technique introduces a ramp-up instead of a pure step for the reference signal.

PI+Lead controller

Figure 5.17 show the performance comparative between analog and digital controller implementations for the PI+Lead compensation expression with $PM = 60^\circ$.

In order to compare disturbances rejection for digital controller and its analog design it is defined a set of periodically changing conditions, observing the converter output voltage for analyze digital control performance.

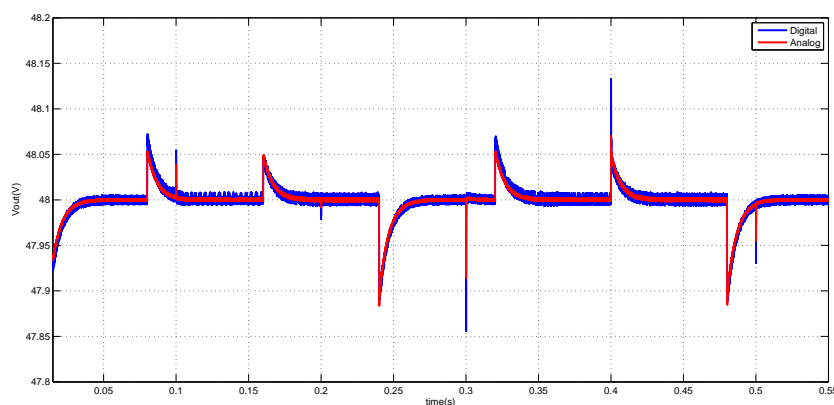


Figure 5.17: Converter output voltage with PI+Lead controller, digital(blue) versus analog(red) compensator comparing disturbance rejection for selected sampling period $T_s \approx 8[\mu s]$.

Test-bench starts with $10ms$ soft-start ramp, with minimum input voltage $V_{in_{min}} = 80[V]$ and nominal load conditions $P_{out} = 500[W]$.

Input voltage is periodically changed from minimum to nominal and maximum condition each $80[ms]$, while output load power changes periodically each $100[ms]$ following the array of values $(500[W], 5[W], 200[W], 1000[W], 5[W])$. This changes evaluate the line and load disturbance respectively.

For example, in figure 5.17 at $t = 80[ms]$ input voltage changes from its minimum $V_{in_{min}} = 80[V]$ up to $V_{in_{nom}} = 110[V]$, after that at $t = 100[ms]$ load power goes from nominal condition $P_{out} = 500[W]$ down to $P_{out} = 5[W]$.

Next figure 5.18 shows more details for the bench-test figure 5.17. On the top can be appreciated a load step from 5[W] up to 200[W] , while at the bottom is shown the step from $V_{in_{nom}} = 110[V]$ up to $V_{in_{max}} = 145[V]$.

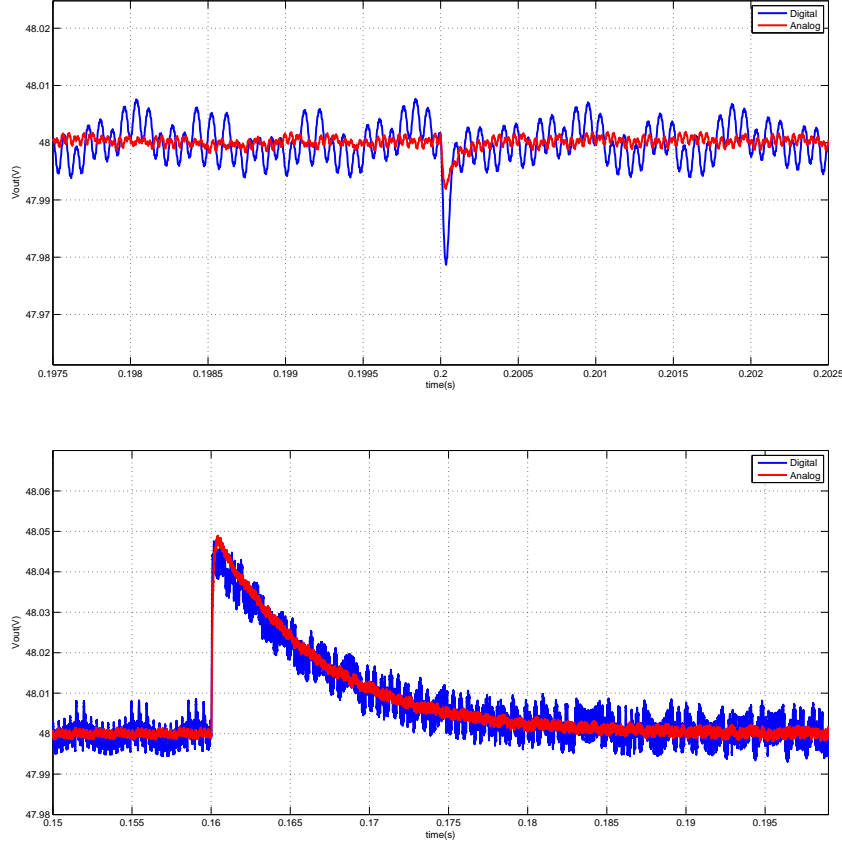


Figure 5.18: Digital and analog PI+Lead controller performance comparison. Top: Line disturbance, input voltage step from $V_{in_{min}} = 80[V]$ up to $V_{in_{nom}} = 110[V]$. Bottom: Load disturbance, output load step from $P_{out} = 5[W]$ up to $P_{out} = 200[W]$

Line and load regulation tracking obtained from digital and analog controllers are really good and practically identical, which confirms that it is possible to use the discretized controller from continuous-time design, under this selected sampling rate.

PID controller

Same test-bench has being applied to PID controller and the result were practically the same, which confirmed again the assumptions of using the discretized analog designed compensator.

The effect of increasing controller computation time t_{ctrl} , and therefore total loop delay t_d , clearly goes in the direction of decreasing the system stability margin. For that reason next sections will study the effect of increasing computation time adding more delay to ADC and DAC sample and hold blocks.

Discretization boundaries

Figure 5.19 shows for same switching frequency how increasing total loop delay affects the stability of the compensator and consequently the perturbation rejection. Compensation algorithm is being recalculated with the respective delayed period.

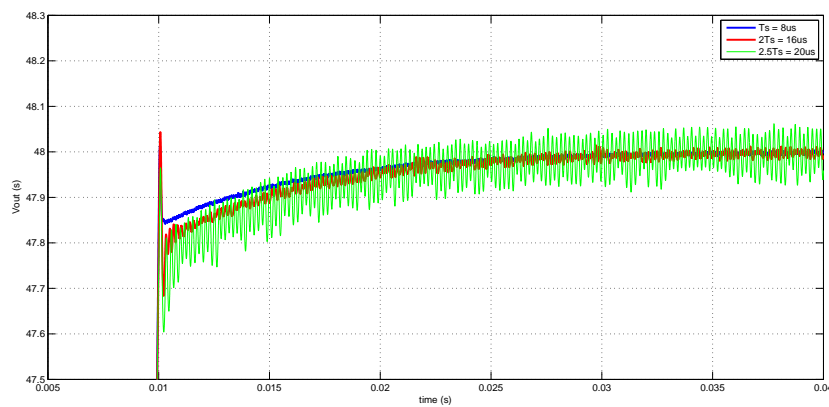


Figure 5.19: Converter output voltage start-up response, increasing controller sample period time clearly drives non-stable oscillating conditions.

As it can be appreciated at the end of soft-start sequence for 2.5 times higher the selected sampling period converter output voltage starts to oscillate and response time is higher due to stability margin reduction. For higher values the system becomes unstable.

Simulations with state space averaged model has been checked and verified that implementation can be done under the conditions explained, despite this in order to get more consistent results, it is being used an electric simulation spice based program called Powersim (PSIM) for validate this simulation results obtained.

5.4 PSIM Simulations

Before starting with the final implementation, it is highly recommended to use spice simulations including the designed controller with its power topology. Doing so, it is

possible to measure the electric magnitudes of each component like the switching MOS-FETs or secondary diodes voltages and currents among other relevant magnitudes.

Powersim is an spice tool specific for power electronics, which is ideal for our application. Using PSIM is possible to use Dynamic Link Library (DLL) block to run a digital code, which is very useful for simulate the implementation of the digital controller. It is a very interesting method to avoid explosions and component damage during the debugging time before real implementation.

In order to be consistent with the results obtained in Simulink®, before simulating the discretized controller several simulations were performed in continuous-time domain, all of them match with the results obtained using the averaged model. This continuous-time domain results are not included because the intention of this section is to know how will affect the compensator discrete time implementation and not to retake analog compensation tests.

5.4.1 Compensation law

In order to simulate the digital compensation algorithm, it is needed to transform the z-domain expressions to time domain.

For example, taking equation 5.2 from PI+Lead compensator, being $u(z)$ the output control signal and $e(z)$ the input error, it is obtained the resultant expression:

$$G_c(z) = \frac{u(z)}{e(z)} = \frac{b_0 - b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} + a_2 z^{-2}} \quad (5.4)$$

Using z transform property:

$$z^{-n} f(z) = Z\{f(k - n)\}$$

And operating terms it can be obtained the compensation law, which describes the output signal value in terms of the input signal and its respective coefficients for each time instant k. Resulting:

$$u(k) = b_0 e(k) + b_1 e(k - 1) + b_2 e(k - 2) - a_1 u(k - 1) - a_2 u(k - 2) \quad (5.5)$$

5.4.2 Simulation scenario

Figure 5.20 show the simulation schematic of the push-pull converter using PI+Lead controller.

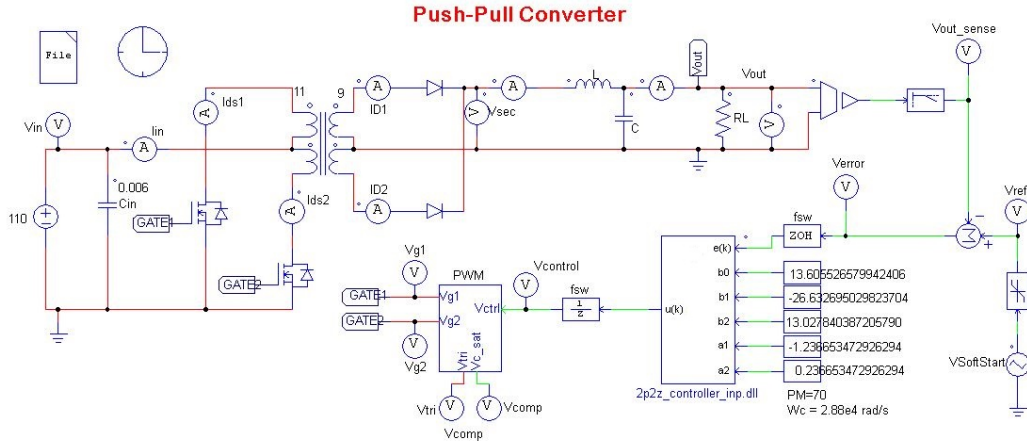


Figure 5.20: Push-pull digital PI+Lead compensator simulation schematic with PSIM.

Push-pull power stage is easy to identify, it is compressed from input voltage source up to output load resistor RL.

Sensing stage is performed with an isolation amplifier, since the final implementation will be powered from primary side. It is followed by a LPF, which is very important in order to get the averaged values from output voltage and to avoid switching harmonic components.

Reference voltage introduces a soft-start sequence like it was done in previous simulation.

To implement the digital compensator the visual studio software has been used to create the DLL block, it computes equation 5.5 for each simulation time step. It has 6 input ports, one for the error signal and the other ones for compensation parameters. Sampling period is a constant internal value. See annexes for check the code used.

To close the loop, PWM block takes the control signal $u(k)$ and gives the gate voltages to its respective MOSFET drivers.

5.4.3 Digital PI+Lead controller

Next figure 5.21 shows with the PI+Lead compensator, the output voltage $V_{out}(t)$ and the compensation signal $V_{comp}(t)$ values during soft-start sequence and after that in steady state under nominal load conditions $P_{out} = 500[W]$.

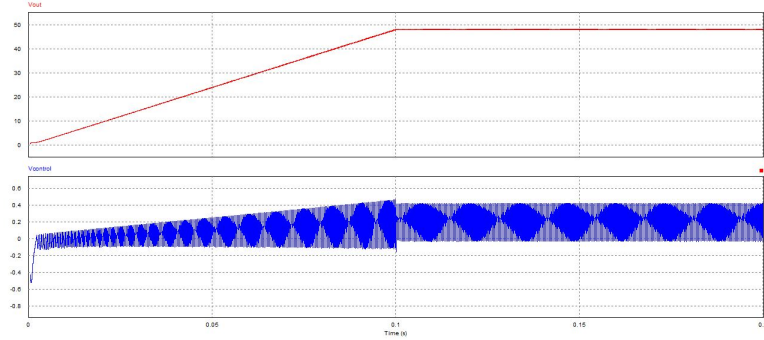


Figure 5.21: PI+Lead digital compensator soft-start sequence, notice that control signal(blue) oscillates in steady state while output voltage(red) oscillations can not be appreciated.

Notice that control signal under steady state became oscillating, this is due to the dynamic response inherit from the controller, which wants to compensate too fast and generates output ripple. See more details in figure 5.21.

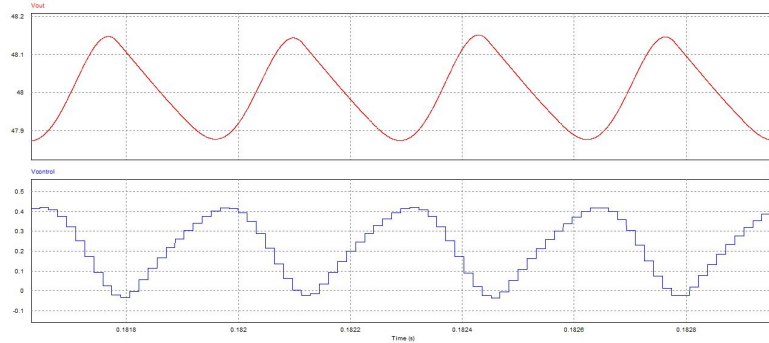


Figure 5.22: Control signal oscillations in PI+Lead compensator in steady state.

Once this issue was detected, there were designed several compensators in order to make the control response slowly and to avoid this aggressive changes. Changing phase margins target designs from 40 to 85 was not possible to reject the oscillatory effect.

5.4.4 Digital PID controller

Applying the same with PID controller under Phase Margin = 60° , the results were much better, control signal do not have any oscillation after the soft-start transient. See next figure 5.23:

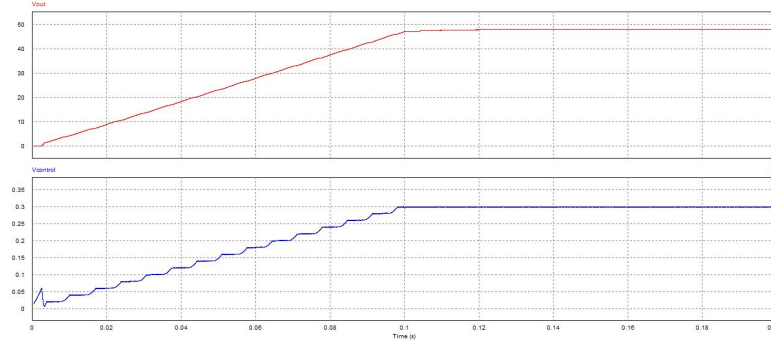


Figure 5.23: PID digital compensator soft-start sequence under nominal output load $P_{out} = 500[W]$.

As expected, the response time is slower and the oscillations disappear.

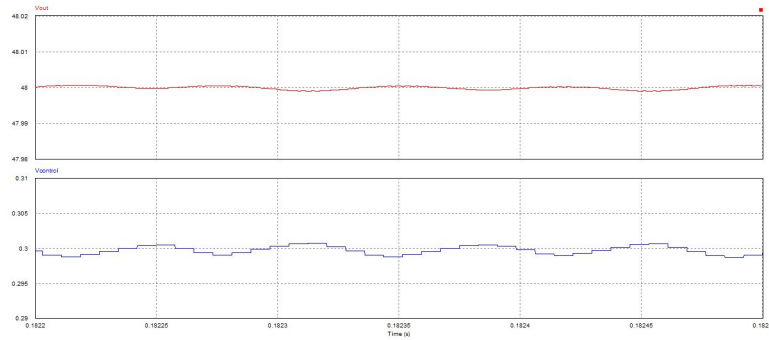


Figure 5.24: PID digital controller output voltage (red) and control signal (blue) after soft-start sequence at $t = 182[ms]$.

Figure 5.25 shows the load disturbance rejection taking a down-step from 100% down to 10% of nominal output power. Notice that response time is $55[ms]$ with an overshoot of 1%.

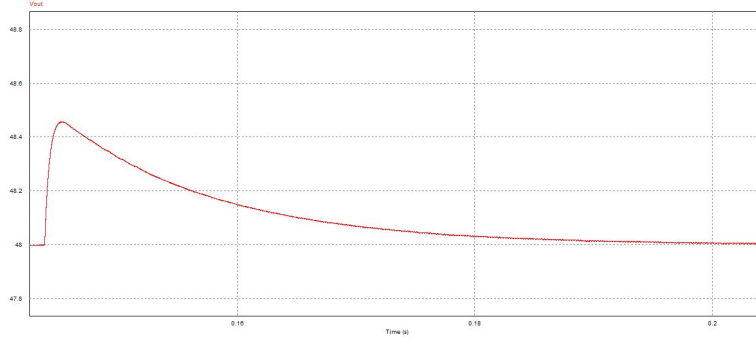


Figure 5.25: Digital PID controller load regulation performance, output power transient from $P_{out} = 500[W]$ down to $P_{out} = 50[W]$.

For line regulation it was performed an step-up from 80[V] to 140[V], resulting 70[ms] transient time and 6% overshoot as figure 5.26 shows:

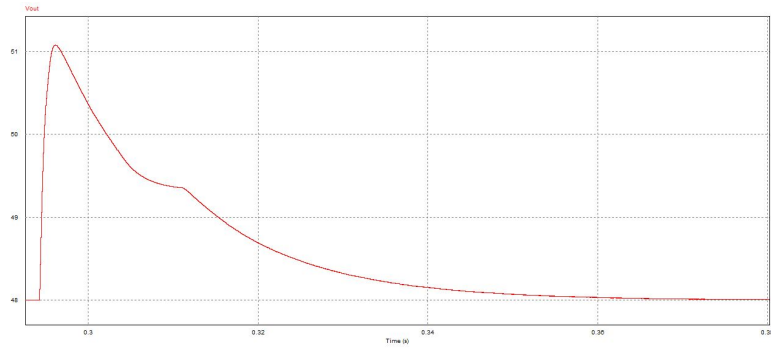


Figure 5.26: Digital PID line disturbance performance, input voltage step from minimum $V_{in_{min}} = 80[V]$ up to maximum $V_{in_{max}} = 140[V]$.

Until now it has been tested the implementation by simulation, selecting PID instead of ZP+Lead controller. The following chapter will explain all the parts required to implement the digital controller in the commercial power supply selected.

Chapter 6

Implementation

To implement a custom digital controller there are a wide range of commercial solutions that it can be divided in three groups; Field-programmable gate array (FPGA), Micro-controller (μC) or Digital Signal Processor (DSP) based on.

In general terms FPGAs are faster than μC or DSPs, because they optimize the hardware management while the development-time effort is considerably higher. This is very important for real products applications where they need fast time to market releases to be competitive, and with this aspect μC or DSP give better results.

DSP is a better solution than μC for digital power because; it runs in higher frequency clock making possible higher switching frequencies, it also makes much faster math operations because μC are oriented to multipurpose applications.

6.1 Firmware

6.1.1 C2000 Launchpad

In order to know the basic performance needed to implement the controller, two questions are being answered;

Is it possible to sample at the frequency selected? For $T_s = 8[\mu\text{s}]$ it is needed a sampling rate greater than $1[\mu\text{s}]$ (1MSPS) in order to have enough time for acquiring and processing.

Is possible to calculate the compensation law with this time? This is directly proportional to the DSP clock latency and the number of computing operations needed. Expected clock frequency should be greater than 25[MHz] for having at least 200 clock periods to implement the calculations for $T_s = 8[\mu\text{s}]$.

For this project it has been selected the Texas Instruments (TI) family C2000 32bits, which it is a mix between DSP and μC , optimized for real-time digital power control, which makes it the ideal option. This family has two subfamilies; piccolo is the low cost and delfino is the high performance one.

In order to speed-up the implementation it is used a launchpad board, it is for two reasons; first is that makes possible to avoid the hardware design of the DSP and second one is TI on-line support is much present and helps to reach the implementation faster. Launchpad boards are perfect for technology testing.

All the families solve the restrictions needed but it is selected the LAUNCHXL-F28379D launchpad board, which is the delfino with highest performance up to date. With this option makes possible to analyze low performance ICs, detect the technological boundaries and select the best choice for cost optimization in the final commercial solution.

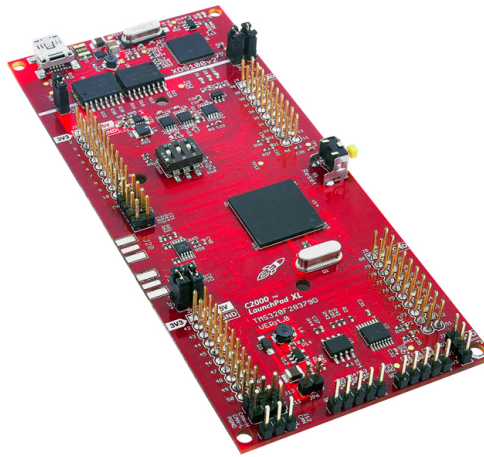


Figure 6.1: DSP selected, LAUNCHXL-F28379D launchpad board from Texas instruments designed for real-time control applications.

The launchpad is composed by a debug and programming circuit, some LEDs and push-buttons, 4 arrays of connection pins with direct access to each port of the main core, TMS320F28379D dual-core μC .

C2000 family is perfect for digital power solution by 3 reasons; it has high-performance core, ADCs and DAC peripherals, see the details listed below [14].

- 32bit floating-point dual-core ($f_{max} = 200[MHz]$)
- High-resolution PWM (resolution 55[ps])
- 4x 16bit ADC at 1.1MSPS or 12bits at 3.5MSPS each.

One of the most interesting benefits from TMS320 cores is 32bits floating-point variables, defined by IEEE 754 standard, it uses 1 bit for sign, 24bits for mantissa and 8bits for the exponential term. It is very important because makes possible to define accurately the coefficients from compensation law, and then obtaining exact values for compensation poles and zeros.

Computation Law Accelerator (CLA) is an independent 32-bits floating-point math accelerator, it has direct access to peripherals enabling parallel execution of the control loop and another independent code. For example in the CLA it can be running the PID compensation law, while in the core is running a routine for send via digital interface the sensed signals.

6.1.2 Program code

LAUNCHXL-F28379D Launchpad board can be programed from Simulink code compiler, adding the closed loop blocks to configure ADC sampling, compensation law calculation and PWM drivers. It makes possible to test the controller in a very simple way. Despite this, it is desired to access core in a lower level to manage resources better, explore in depth the solution and adapt it to the particular case.

Code Composer Studio (CCS) is an Integrated Development Environment (IDE) to develop TI embedded processors like TMS320 based on eclipse open source IDE. It makes possible to program the code and debug in real-time the variables using JTAG probes like XDS emulators. For F28379D core it is needed CCS version 6 or later.

Programming language used in LAUNCHXL-F28379D is C, what makes more easy to program the device because it is the most common language for MCUs.

Bloc diagram

The program starts running when the converter input voltages are applied because it is powered from the primary side, then it starts when power up occurs. Figure 6.2 explain the implemented code-flow running inside the DSP.

Once the DSP is powered, it runs the set-up and initialization code; setting General Purpose Input Output (GPIO) ports, Analog to Digital Converter, Pulse Width Modulation registers and also each Interrupt Service Routine (ISR).

Soft-start function increases progressively the reference voltage from 0 up to the nominal value in function of ramp time, which can be configured from 1[ms] up to 10[s]. It is needed to drive the converter up to its steady state output voltage value and avoid

compensation signal saturation during start-up sequence.

When sequence ends, it is checked if output voltage is inside the maximum and minimum acceptable values and if it is not the case, it re-takes soft-start sequence once. If the next sample is again wrong, it pauses the converter by setting null duty cycle and calls the error check function.

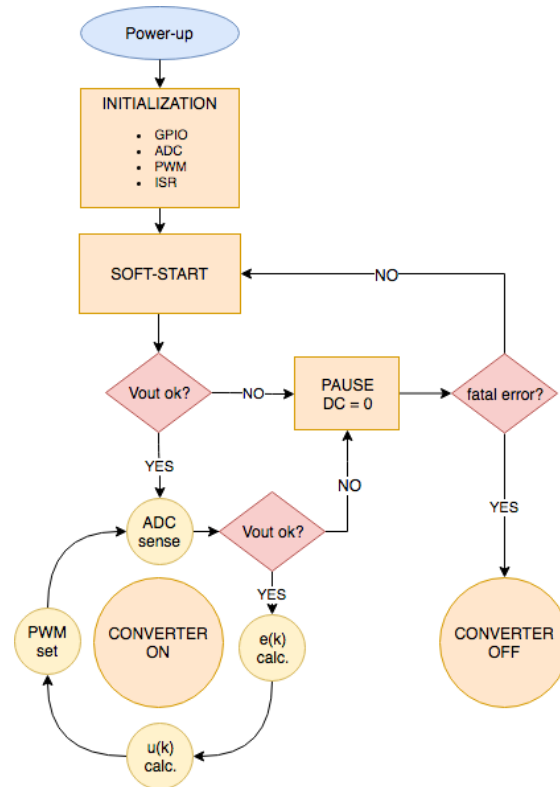


Figure 6.2: Push-pull converter firmware bloc diagram for Voltage Mode Control implementation.

Error check function measures the input voltage and current, then decides if it is needed to stop permanently the converter or avoid suspension and run the soft-start sequence again.

The main loop of the code is in charge of sensing the output voltage, check if output voltage is as expected, calculate the compensation law and actualize the PWM values each period time. It is done by an Interrupt Service Routine because it forces maximum priority of execution.

Converter ON routine is executed as an infinite loop in parallel with the controller ISR. If everything is going well, LED2 is blinking constantly and input magnitudes (voltage and current) are being sensed each second.

6.2 Hardware

Implementation is being performed using a commercial power converter as it was explained in initial sections. This part explains how each hardware modification of the CRS-500 converter is done and why.

Before modifying the converter it was analyzed deeply in order to understand which sections will need modifications or replacements. The converter can be divided by four main parts; power stage, sensing, control and alarms. Power push-pull stage is not modified, each part is fully original.

In original analog controller, compensator transfer function is done by classic techniques with a set of resistors and capacitors followed by the respective operational, while the PWM generation and the control itself is done with an integrated chip. Then, controller will be obviously removed and replaced by the Launchpad board.

Finally it is easy to demonstrate that changing the controller, sensing will need to be adapted also because DSP is powered at a different voltage level.

First step of implementation is to control the power stage under open-loop conditions. Then it is needed to start by introducing the digital controller into the PWM converter loop.

6.2.1 Compensator

Controller used by CRS-500 belong to the family of general purpose PWM modulators UCx52xA from TI. These ICs, are very popular in power electronics applications because they offer control solution for a large number of DC/DC topologies.

Some relevant features are: adjustable switching frequency, wide supply voltage range, dead-time control, internal soft-start, current limit, integrated error amplifier and voltage reference. For all these reasons and with a very good market price (less than 1€), the UCx52xA family is a very good solution for DC/DC converter analog control.

For controller hardware, only few modifications are done because LAUNCHXL-F28379D board integrates all the needed parts. Two changes are done, first of all removing each part related with the analog controller UCx52xA and then adapting DSP supply voltage from analog controller previous level.

Controller can be powered from primary or secondary side but normally it is done in the primary side for avoid secondary transformer windings. Selected converter has a linear

regulator that reduces the input voltage down to $V_{cc} = 12[V]$.

LAUNCHXL-F28379D needs $V_{dd} = 3.3[V]$ and DSP core maximum current consumption is $I_{DSP} = 0.45[A]$, taking in care the rest electronic components of the board, the estimated maximum current consumption is $I_{dd} < 0.8[A]$ [14].

To regulate the supply voltage it is selected the low-dropout voltage regulator TLV1117-33 from TI. Next figure 6.3 shows its electric schematic:

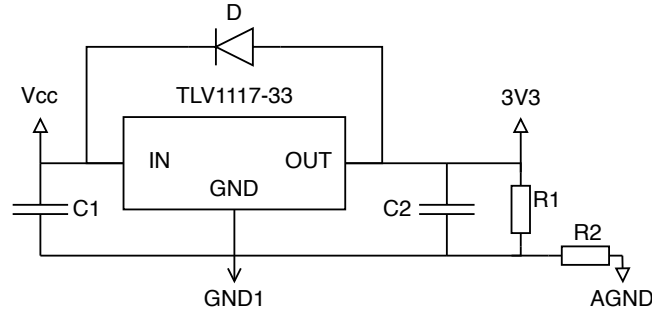


Figure 6.3: Low-Dropout voltage regulator that supplies power to digital controller launchpad board.

Capacitor $C1 = 10 [\mu F]$, and $C2 = 100 [\mu F]$ are the highest values that voltage regulator supports before becoming unstable. Higher capacitances help to reduce power supply ripples.

Notice that diode D protects from input voltage V_{cc} short-circuits, R1 is a small charge of $10 [mW]$ to reduce voltage ripples under light-load conditions and R2 is a resistance to connect sensing analog ground (AGND) to supply voltage ground (GND1).

6.2.2 MOSFET Driver

Once the launchpad is powered, next step is to drive the switching power MOSFETs in order to get an un-regulated output constant voltage. For that reason is was developed a firmware to generate PWM signals at switching frequency of $124 [kHz]$ with selectable duty cycle from the debugging CCS environment interface. Duty cycle from each MOSFETs is being restricted from 0% up to 45% in order to avoid shoot-through.

Drivers were included inside analog compensator but now it will be needed to add a driving stage. Power MOSFETs threshold voltage range is $2[V] < V_{gs(th)} < 4[V]$, which can be achieved by the DSP power supply, but to avoid signal integrity problems like erratically conduction it is common to work with higher voltages. In this case $V_{cc} = 12[V]$,

what makes ideal for supply transistor gate-source V_{gs} voltage drivers.

To convert PWM with $V_{dd} = 3.3[V]$ amplitude to $V_{cc} = 12[V]$ it is used the dual non-inverter driver UCC37324D from TI. Figure 6.4 show the schematic used to drive converter transistors.

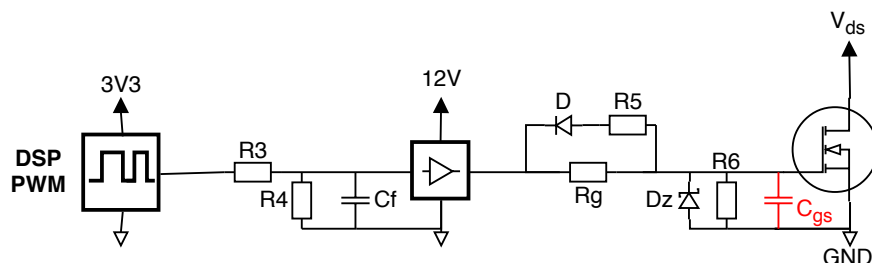


Figure 6.4: Power MOSFETs driving stage schematic. Notice that C_{gs} is not a real capacitor because it is a parasitic component from transistor.

Resistor $R4 = 10 [k\Omega]$ and protects spontaneous driving when DSP PWM block is disconnected and some conducted or radiated noise is introduced on driver input. Resistor $R3 = 1 [k\Omega]$ and capacitor $Cf = 100 [pF]$ are a Low-Pass Filter (LPF) to avoid high-frequency ringings and noise injected to drivers. Filter components are selected by experimental tuning, the cleaner V_{gs} waveform gives a LPF bandwidth defined by:

$$f_c = \frac{1}{2\pi R_3 C_f} = 1.59 [MHz]$$

Ideal driving stages do not take care about time delays between generated DSP and received signals (power MOSFETs) but in real implementation it is a very important aspect. There are multiple factors that introduce delays in the chain, it can be divided in three parts; propagation delays between input and output, slew rates from generators and loads from receivers.

While PWM period time is $T_{sw} = 8 [\mu s]$, propagation delays and slew rate times are in the range of few nanoseconds, which represent an error less than 1%.

The most important delay came from power MOSFETs input capacitance. Miller theorem introduces that input capacitor can be divided in two parts, the most important in this case is the gate to source C_{gs} capacitance, see figure 6.4. The time needed to drive the signal is defined by the charge time of this capacitance.

R_g resistance limits the current that charges C_{gs} while resistor R5 and D diode limits the discharge current. Resistance values were obtained by experimental tests, concluding

that $R_g = 15 [\Omega]$ and $R_5 = 1 [\Omega]$.

To protect gate over-voltages is used the zener diode Dz with $V_{th} = 18[V]$, while resistor $R_6 = 1 [k\Omega]$ acts as a driving protection in case of MOSFET failure.

Under nominal conditions of input voltage, load charge and duty cycle it was possible to obtain the nominal output voltage and also modifying output voltage value by means of CCS software debugging.

This validates open-loop, next steps clearly go to obtain feedback, and for that reason it is needed to sense the magnitude in order to close the control loop.

6.2.3 Sensing

To implement Voltage Mode Control technique it is only required to sense output voltage but for detecting possible incidents, input current and voltage will be also monitored. Input current have special interest in case of using other control techniques like Averaged Current Mode Control (ACMC) because it is proportional the inductance current.

In digital control, under small ripple approximation and averaged values it is not important to sense high-frequency components like output voltage and inductor current ripples, in fact only the DC components are desired.

LAUNCHXL-F28379D has 4 channels with independent analog-to-digital converter with programmable resolution of 16-bits or 12-bits, each of them are capable to acquire at least at 1.1MSPS with 3.3 [V] of dynamic range. In order to define the measuring ranges and resolution needed, it have been used the maximum theoretical values for each magnitude with an additional margin, following table 6.1 summarizes the ranges and resolutions that can be achieved:

Table 6.1: Sensed magnitudes summary

Magnitude	Range	12b ADC	16b ADC
V_{out}	0..65 [V]	15.86 [mV]	0.99 [mV]
V_{in}	0..145 [V]	35.4 [mV]	2.21 [mV]
I_{in}	0..6.5 [A]	1.58 [mA]	0.09 [mA]

If 12-bits ADC is used the resolution error is $e_{12b} = 240[ppm]$ while using 16-bits it is reduced to $e_{16b} = 15[ppm]$, but in practice sensing noise will be higher than 12-bits resolution error. Needed quantizing time will be less for 12-bit converter. Then, ADC resolution selected is 12-bits for output voltage V_{out} , input current I_{in} and voltage V_{in} .

Output voltage

The converter has two sensing terminals to compensate voltage dropouts in cables, which are perfect for sensing the converter output voltage V_{out} .

Push-pull is an isolated DC/DC topology and DSP is powered in primary side, in order to guarantee the isolation between both transformer sides output voltage sense will need to be isolated. This can be done using an isolation amplifier, selected IC is HCPL7800A from Avago.

The isolation amplifier needs 5 [V] at each side, in the primary side it can be solved using the L7805 from ST, linear low-dropout regulator connected to $V_{cc} = 12[V]$ with fixed output voltage of 5 [V]. In the secondary side it is done with the isolated mini DC/DC converter NMV0505SAC from muRata.

Isolated resultant voltage is differential but as the selected 12-bits ADC is single-ended referenced to analog ground (AGND), it is needed an amplification stage to adapt it. To avoid protections from 5 [V] to 3.3 [V] this amplification stage works at the same voltage as ADC, that can be done using a reel-to-reel operational amplifier, selected IC is LMC6482 from TI which has only 20 [mV] supply rail.

Figure 6.5 is the block diagram for the electric schematic of the output voltage sense. As it can be appreciated output voltage is filtered and then attenuated, is important to filter as close to the signal as possible in order to avoid noise amplification in the following stages. LPF cut-off frequency is $f_c = 5[kHz]$, it is for attenuating all switching harmonic components.

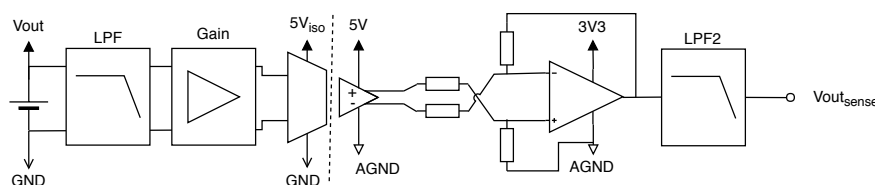


Figure 6.5: Output voltage sensing schematic bloc diagram.

LPF2 stage eliminates possible noises introduced in the sensing chain, LPF2 cut-off frequency is $f_c = 72[kHz]$. Both filters are passive first order RC, LPF2 has more bandwidth because for practical reasons it is not recommended to connect high impedance in series with the ADC, doing so introduce load charging errors.

Figure 6.6 shows the implemented module used for output voltage sense:

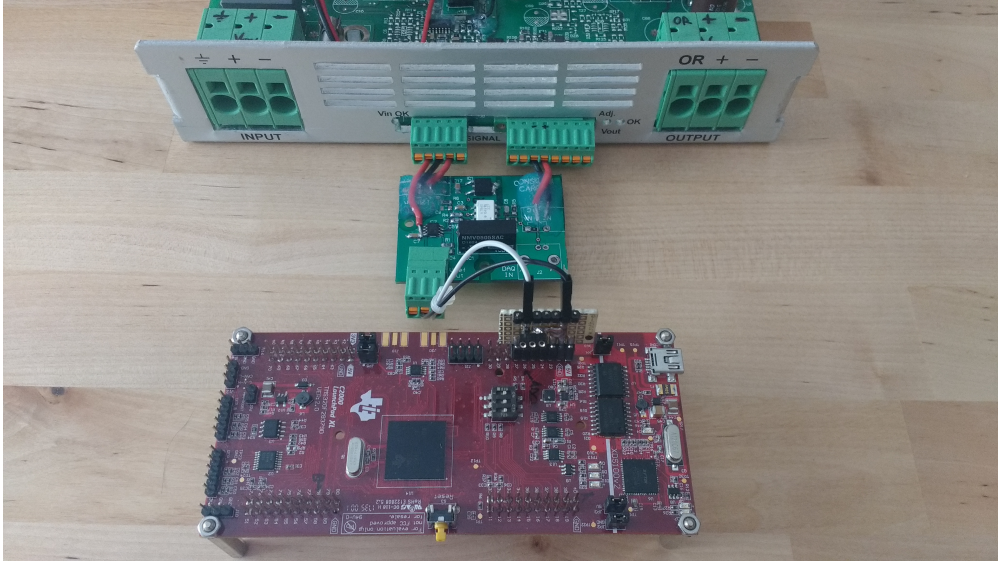


Figure 6.6: Output voltage sensing board implementation connected from power converter to LAUNCHXL-F28379D (DSP).

Input current

Measure output inductance current is better than input current, but it is difficult to include a sensor between the transformer and the output LC filter. It is because as a design rule, the path between transformer and LC filter is minimized to reduce radiated and conducted emissions effects.

Then the alternative is to measure the primary side current which is directly related with the inductance current and has more space for route the current sensor circuit. Non-ideal transformers have leakage currents that introduce coupling coefficients $k < 1$, being $I_L = kI_{in}$. See reference [2] section 13.2 for more details.

The better way to measure input current is using a current transformer and a transconductance from current to voltage. It is an isolated measurement which makes easier the reference routing, only needing a few passive components and has very good accuracy results.

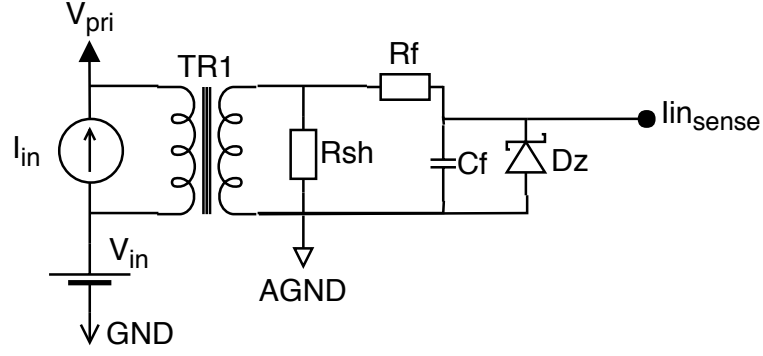


Figure 6.7: Input current sensing schematic bloc diagram.

Figure 6.7 show the block diagram for input current sense. TR1 current transformer used is PE-67100NL from Pulse Electronics, with turns ratio of 1:100 is capable to measure currents up to 37 [A] with an operational range is from 10 [kHz] up to 200 [kHz].

Secondary current is trans-conducted to voltage using a shunt resistor, its value is selected for having maximum voltage at full scale range. The maximum input current is calculated for minimum input voltage of $V_{in_{min}} = 80[V]$ under full load conditions $P_{out} = 500[W]$.

$$R_{sh} = \frac{V_{sh}}{0.01I_{in}} = \frac{3.3[V]}{0.065[A]} = 50.7 [\Omega]$$

As the current transformer works only for alternated signals and the desired value is the DC component a LPF with $f_c = 1.59[kHz]$ being resistor $R_f = 1 [k\Omega]$ and capacitor $C_f = 100 [nF]$.

Input currents larger than 6.25 [A] will damage ADC because it will generate a signal greater than supply voltage, for that reason it is used a transil diode with $V_{th} = 3.3 [V]$ to clamp higher voltages. Selected IC is SMLVT3V3 from ST.

Input voltage

To sense the input voltage it is used a resistive voltage divider and a reel-to-reel operational amplifier LMC6482 working as buffer stage. Figure 6.8 shows the schematic:

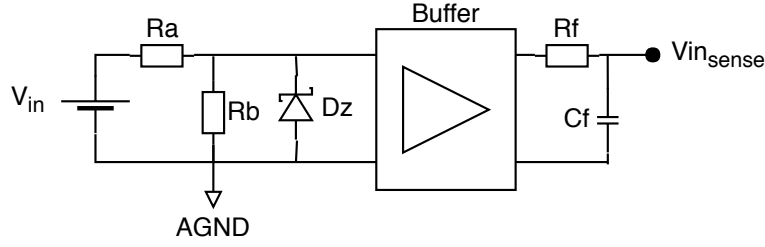


Figure 6.8: Input voltage sensing schematic.

Like in current measurement, the voltage can be higher than supply level and it is needed to add a transil diode. LPF bandwidth is exactly the same as current sense. Voltage divider is designed for having the supply voltage with the maximum sensing value, being:

$$A_v = \frac{V_{sense_{max}}}{V_{in_{max}}} = \frac{3.3[V]}{145[V]} = 0.02275$$

Fixing $R_a = 100[k\Omega]$ limits the transil diode current to $I_{Dz} = 33[\mu A]$ and also limits the voltage divider dissipation power to $P_{diss} < 210[mW]$.

$$I_{Dz} = \frac{V_{th}}{R_a} = \frac{3.3[V]}{100[k\Omega]} = 33[\mu A]$$

$$P_{diss} = \frac{(V_{in})^2}{R_a + R_b} = \frac{145^2[V]}{100[k\Omega] + R_b} < 210[mW]$$

Taking the expression of gain results resistance $R_b = 2.328[k\Omega]$.

$$A_v = \frac{R_b}{R_a + R_b} \rightarrow R_b = \frac{A_v R_a}{1 - A_v} = \frac{0.02275 * 100[k\Omega]}{1 - 0.02275} = 2.328[k\Omega]$$

All resistors used for sensing purposes are 1206 SMD package with 0.1% tolerance (E192 series), taking the nearest direct values or equivalent parallel combinations. Capacitors are ceramic with also 1206 SMD package.

Chapter 7

Experimental Results

Power converters have some usual technical specifications, some of them are directly related with the control technique while other results are defined by the hardware itself. In this section it is reported all the laboratory testing assessment done for validate the controller design.

Full implementation test was done by progressively adding the needed parts. First of all checking the MOSFET drivers, continuing with the sensing magnitudes and finally adding the closed loop. All the laboratory work done during those steps are not included in the following experimental results.

7.1 Bench-test definition

To obtain the data from those experiments it was needed some instruments and tools listed above:

- Personal Computer: MSI CX61-2QC
- Power Supply: EA-PSI 9200-140 3U
- Loads: Passive self-manufactured and ITECH IT8512C (electronic load)
- Oscilloscopes: Yokogawa DLM2054 and Hantek DSO5102P
- Current probe: Fluke PR430
- Digital Volt Meter: Uni-T UT136B and Fluke 175

Figure 7.1 shows the self-manufactured output load, needed for doing the following tests. Notice that the switches add or withdraw parallel resistances directly to the input terminals of the load.

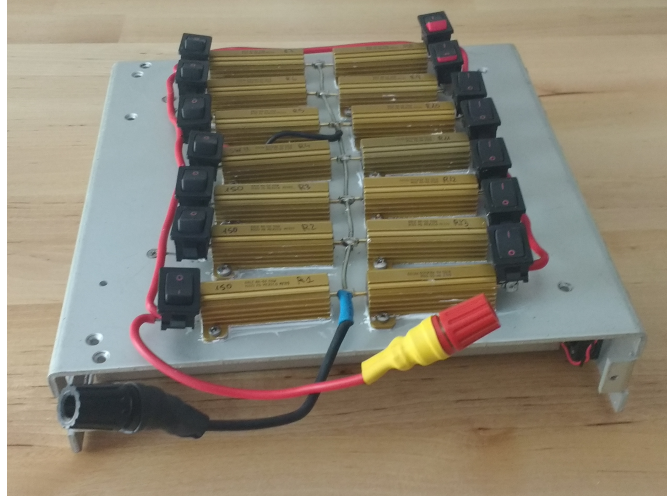


Figure 7.1: Passive load array used for Device Under Test output power charge variation.

7.2 Load regulation

This measures how well the converter is able to reject the output load variability, which is directly related with the controller performance. Most manufacturers give this specification by the worst case, being it the output voltage variation from 10% to 90% of the nominal converter power.

To calculate load disturbance rejection, it is used the power supply at nominal input voltage of 110[V] and with a current limit of 8[A] for avoid transient effects. To obtain the multiple output power it is used the resistive load, by connecting in parallel more resistors it is possible to get 14 different power states.

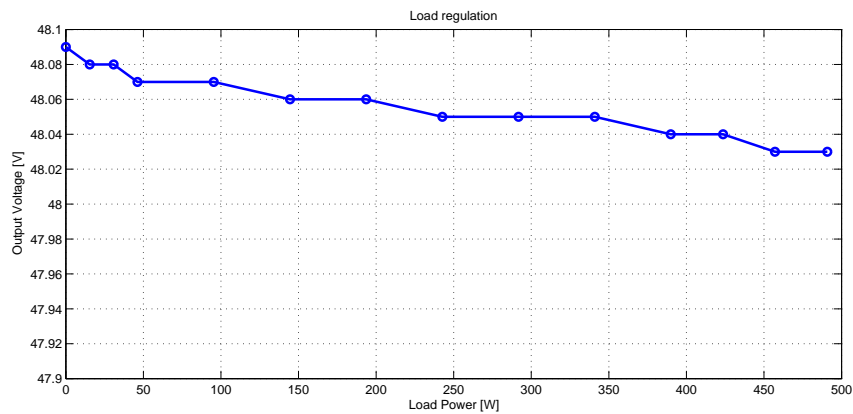


Figure 7.2: Output voltage sensed varying load charges give the load regulation curve of the converter.

To calculate converter output load power it have been measured the resultant parallel impedance with Fluke 175 DVM and by ohm law derived from the output voltage measured during the test. Figure 7.2 includes the output voltage value for each output power range.

The resultant load regulation in the worst case can be calculated as:

$$LOAD_{reg} = \frac{V_{max} - V_{min}}{V_{max}} * 100 = \frac{48.09 - 48.03}{48.09} * 100 = 0.125[\%]$$

Manufacturer gives a load regulation $< 0.2\%$, which is consistent with the result obtained.

7.3 Line regulation

Line regulation is also directly related with the controller performance, it measures how good is the input voltage disturbance rejection for the power converter under test.

The test is very similar than the load regulation but in this case the output needs to remain in nominal conditions with output power of 500[W] while the input voltage is being changed from 80[V] up to 140[V]. The input power supply current limit is set to 8[A] again. Input voltage is measured using UT136B DVM while for output voltage it is used the Fluke 175 DVM.

Figure 7.3 give the converter output voltage sensed for obtaining the line regulation.

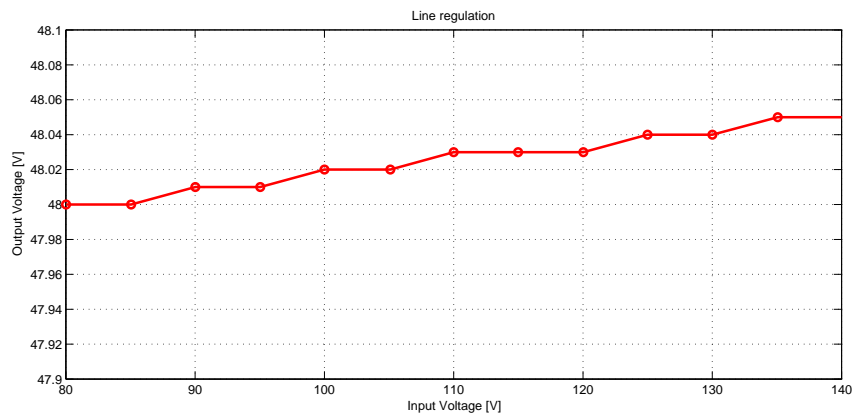


Figure 7.3: Output voltage sensed varying input supply voltage give the line regulation curve of the converter.

The resultant line regulation in the worst case can be calculated as:

$$LINE_{reg} = \frac{V_{max} - V_{min}}{V_{max}} * 100 = \frac{48.05 - 48.00}{48.05} * 100 = 0.104[\%]$$

Manufacturer gives a load regulation $< 0.2\%$, which again is consistent with the result obtained.

7.4 Efficiency

Probably it is the most important parameter for a power converter but it is not related with the controller performance because it is measured in steady state conditions and it quantifies the non-ideal losses from the electronic and magnetic components.

To calculate efficiency it is needed to measure input and output power. For input terminals it has been used the current measurement given by the power supply instrument while the voltage is taken from UT136B DVM instrument. The output power is measured by the output voltage and derived from the equivalent resistance measured previously, like in load regulation test.

Figure 7.4 shows the converter efficiency curve:

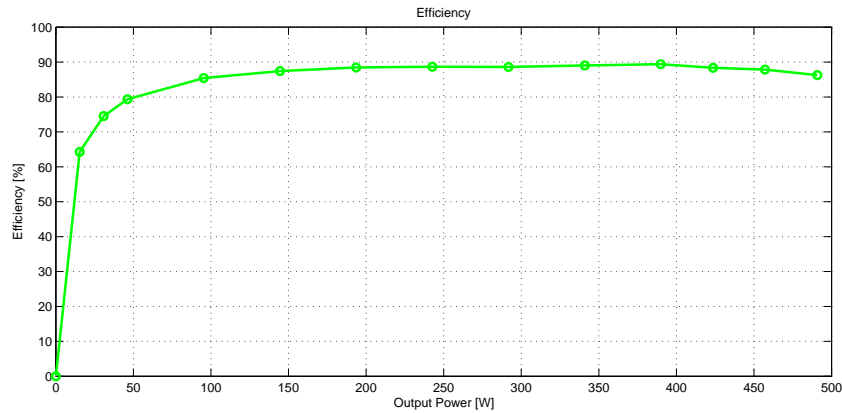


Figure 7.4: Efficiency curve obtained from multiple output power values.

Manufacturer give an efficiency value of $\eta = 92\%$, while from results obtained the best case is $\eta = 89.6\%$. Digital controller and sensing electronic components power consumption is $7.8[W]$, which represents 1.56% of the efficiency and then this deviation can be attributed to this parts added. Part of the resting 1% deviation provably came from sensing equipments used, which do not have extraordinary accuracy because they are low cost.

7.5 Ripple and Noise

Output voltage ripple gives good information about the stability of the converter under nominal conditions because if the compensation signal oscillates it can be appreciated in the output voltage ripple measurement. Most manufacturers give this parameter taking into account the noise or they give both measurements.

The most common is to measure the output voltage ripple and noise with an oscilloscope and limiting the acquisition bandwidth to 20[MHz]. For measuring the ripple it is very important to take the measurement as close to the output terminals as possible and create a short loop from positive to reference terminals of the sensing probe.

Figure 7.5 shows the ripple and noise measurement obtained, being $\Delta V_{out} = 100[mV_{pp}]$ and $noise = 720[mV_{pp}]$:

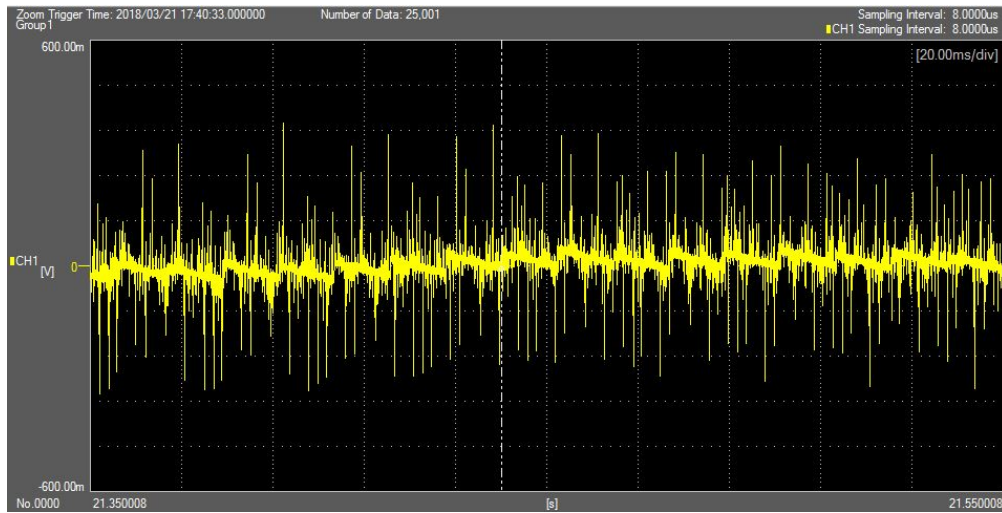


Figure 7.5: Ripple and noise measurement at nominal input voltage and 1[A] output load, 20[MHz] acquisition bandwidth, AC coupling, horizontal scale = 20[ms/div], vertical scale = 100[mV/div].

If it is analyzed in depth, output voltage has an oscillation each 12.5[ms] as it can be appreciated in figure 7.6 due to compensation effort. This issue plus other low frequency oscillations from unknown origin produces more ripple than expected, it provably comes from the output voltage sensing noise.

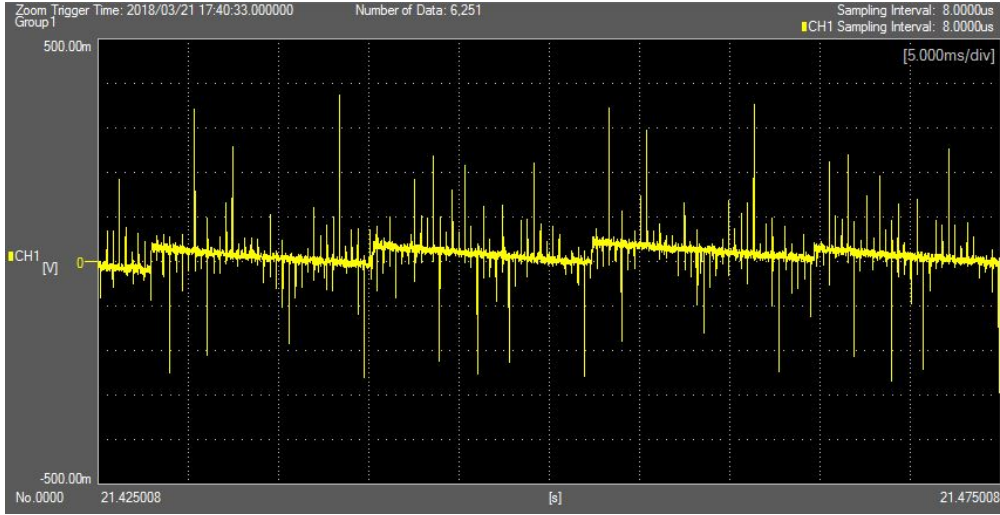


Figure 7.6: Ripple and noise detail measurement for notice output voltage oscillation. BW=20[MHz], AC coupling, horizontal scale = 5[ms/div], vertical scale = 100[mV/div].

Manufacturer specifications are $\Delta V_{out} = 50[mV_{pp}]$ and $noise = 100[mV_{pp}]$ which clearly differ from the measured values. Integrating the DSP in the same Printed Circuit Board sampling noise will be reduced drastically and ripple performance will be improved.

7.6 Transient response

In order to know the behavior of the controller under transient changes it was not possible to use the passive load for doing abrupt changes and for that reason it was used the electronic load ITECH IT8512C, working in constant current mode.

Figure 7.7 shows the output voltage measured with DLM2054 oscilloscope for a transient current step-up from 0[A] to 5[A]:

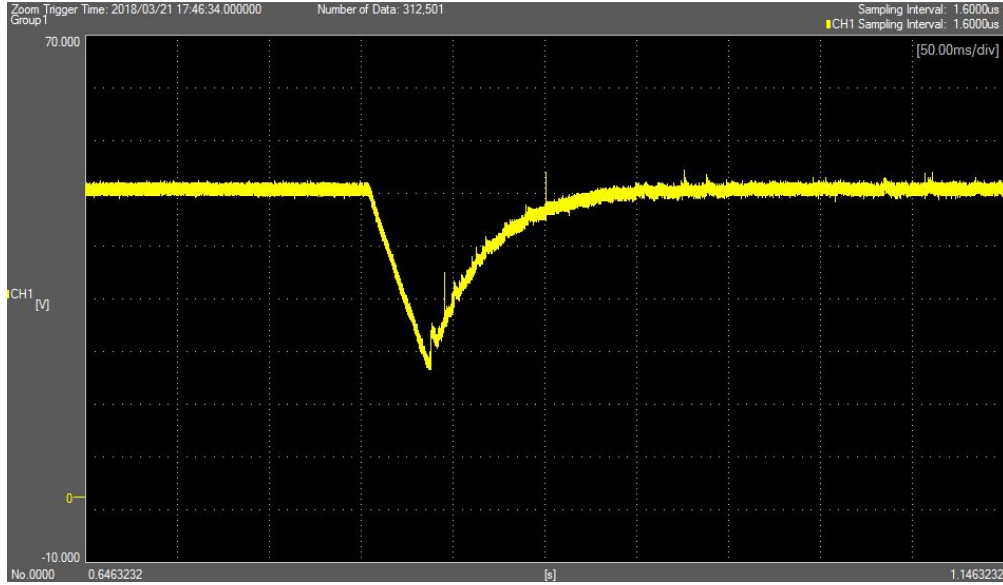


Figure 7.7: Output voltage transient response under 5[A] load current step-up. BW=20[MHz], DC coupling, horizontal scale = 50[ms/div], vertical scale = 8[V/div].

With such load step the output voltage is drastically decreased to 26[V] during 31[ms] and after that it is compensated to nominal output voltage in 58[ms] time. The response before compensation has a linear behavior due to electronic load working principle and for the compensator bandwidth which is quite slow. Compensation time is only associated to the designed compensator, which means that no matters how far is the reference signal, the time response must be always the same.

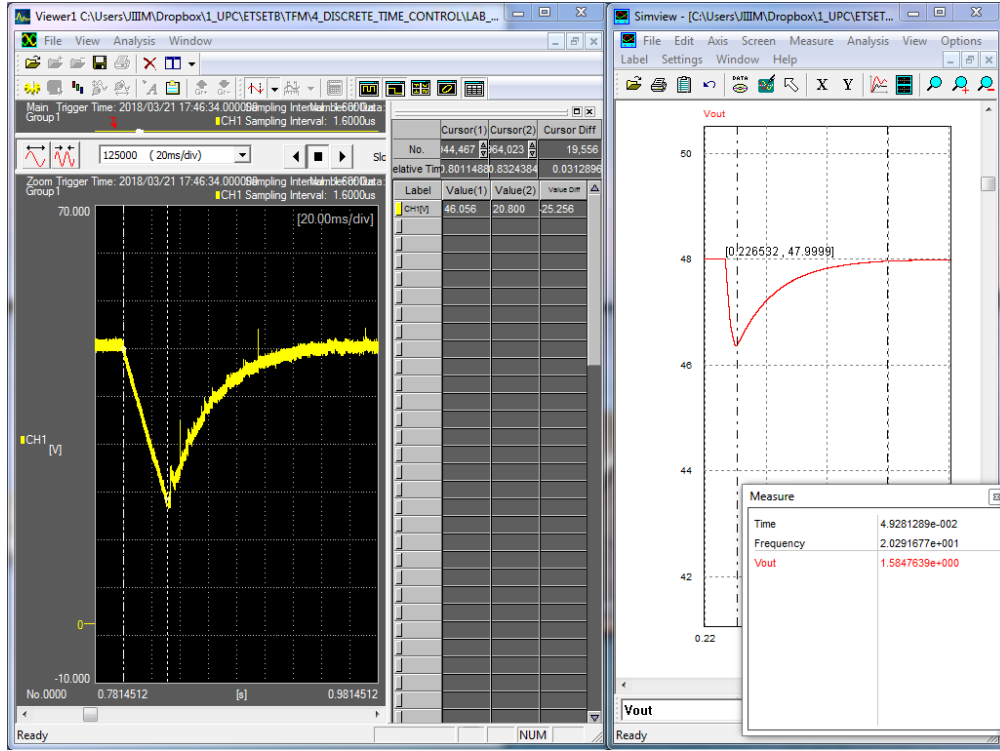


Figure 7.8: Output voltage transient response comparative between experimental and PSIM simulation results for 5[A] load current step-up.

By PSIM simulations, it have been obtained similar values for the compensation time. The valley takes 5[ms] while compensation needs 50[ms] to compensate 1.6[V] and reaches 48[V] again. Figure 7.8 shows the results obtained in order to justify the electronic load undesired response produced.

7.7 Soft-start

The controller design is done under nominal conditions of input voltage and output load. In order to avoid compensation signal saturation, the converter is driven to nominal conditions and then starts to work the compensation law in closed loop for maintaining the converter regulated.

In other words, the converter has an start-up sequence known as soft-start. Ideally it should take 100[ms] ramp-up time but in order to observe it better, time is set to 10[s]. Figure 7.9 shows the output voltage measured with DLM2054 oscilloscope.

Notice that at the beginning there is a damped step, it is because the minimum duty cycle is limited by software in order to avoid overflows in duty calculation. If we trace a

line with the ramp-up slope we can see that start time is 1 second early.



Figure 7.9: Output voltage soft-start 10[s] sequence. BW=20[MHz], DC coupling, horizontal scale = 1[s/div], vertical scale = 10[V/div].

At the end of the sequence, the output voltage is not 48[V] yet, in fact it is around 40[V] and it can be appreciated when the controller starts working. It give again a transient time response of 50[ms] like figure 7.10 show:

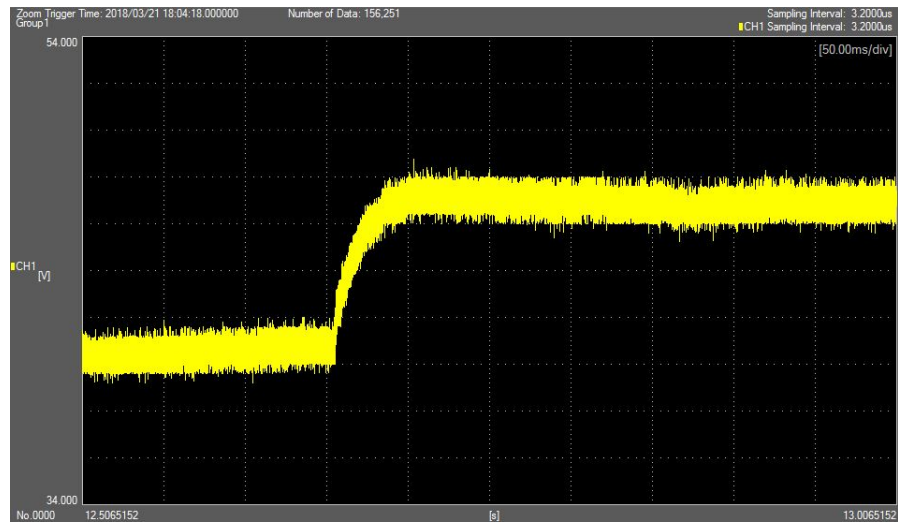


Figure 7.10: Output voltage transient response after soft-start sequence ends and controller start running. BW=20[MHz], DC coupling, horizontal scale = 50[ms/div], vertical scale = 2[V/div].

Chapter 8

Budget

8.1 Design, prototype and implementation costs

In order to calculate a realistic value of the investment needed, next scenario is posed.

The Electronic engineer is a freelance with a working office rented, the annual salary is 60,000€/year and with 35% freelance tax, working during 6 months in the project.

Instrumental and software costs will be amortized in 2 year, so only 25% will be charged to the final cost. Costs that are imputed to the project itself are fully charged.

Salary (100% charged):

SubTotal: 19,500.00€

Work zone (100% charged):

- Rental office fee: 2,100.00€(350€/month)
- Office material: 154.00€
- Laboratory tools: 253.00€

SubTotal: 2,507.00€

Software (25% charged):

- Matlab/Simulink: 2,000.00€
- PSIM: 5,000.00€
- Altium Designer: 495.00€(Circuit Studio)
- CCS: free
- Visual Studio: free

SubTotal: 1,873.75€

Instruments (25% charged):

- MSI CX61-2QC (PC): 725.00€
- EA-PSI 9200-140 3U (PSU): 7,085.00€
- Passive Load: 43.28€
- ITECH IT8512C (EL): 730.74€
- Yokogawa DLM2054 (OSC): 9,320.00€
- Hantek DSO5102P (OSC): 276.00€
- Fluke PR430 (C.Probe): 256.00€
- Fluke 175 (DVM): 229.00€
- Uni-T UT136B (DVM): 21.09€

SubTotal: 4,671.50€

TOTAL: 28,552.25€

8.2 Prototype BOM

CRS500 power converter was a present from Premium company, thanks again. The costs are not included in the prototype budget.

This implementation has taken profit from the converter PCB layout for welding some components, the DSP used is already routed because it is a launchpad board and the output voltage sensing PCB is recycled from previous projects. In summary, the prototype do not use any PCB, only needs electronic components.

Next list indicates the cost from components needed to realize the prototype:

- 22x SMD 1206 capacitors: 0.924€
- 2x SMD 0603 capacitors: 0.072€
- 12x SMD 1206 0.1% 0.25W Resistor: 1.272€
- 29x SMD 1206 1% 0.25W Resistor: 1.450€
- 6x diode LL4148-GS08: 0.066€
- 2x diode BYG10M-E3/TR: 0.148€
- 1x DCDC mini-converter NMV0505SAC: 4.680€
- 1x LAUNCHXL-F28379D: 28.700€
- 5x zener BZX84C15LT1 : 0.040€

- 4x fuse SMD 1206 MF-MSMF050-2: 0.308€
- 2x OpAmp LMC6482AIM : 1.434€
- 1x PWM driver UCC37324D : 0.768€
- 1x Voltage reference TL431CPL: 0.068€
- 1x LDO Voltage regulator LM1117-33: 0.333€
- 1x LDO Voltage regulator LM1117-50: 0.333€
- 4x 10pins 2.54 pitch connector: 0.120€
- 1x Isolation amplifier HCPL-7800: 4.520€
- 1x Transistor IRFI740G: 0.506€
- 1x Transistor BUT11AF: 0.357€
- 1x Transistor BC807-25: 0.016€
- 1x Current transformer PE-67100NL: 1.480€

TOTAL: 47.60€

8.3 Financial viability analysis

Final product must integrate the DSP and sensing circuits in the converter PCB in order to use the same mechanic enclosure.

Some components used in the implementation are also needed by the initial product, then the costs can not be attached to this solution and they are not taken in care; for example PWM drivers , input voltage and current sensing circuits. Main difference between this prototype and final solution is the DSP cost, Launchpad cost is 28.7€ while TMS320F28379D IC bought alone costs 17.24€. If those considerations are applied to prototype expenses the final solution can be achieved from 32€.

The converter selected have a list price from 500€. With the proposed implementation of the digital controller it will be possible to improve the services given to the tentative clients, taking benefits from the data monitoring and control algorithms complexity. In the other hand, design and development cost can also be reduced because the control platform hardware do not need any change.

It is clear that the digital controller introduces more expenses, being 6.4% from the converter list price, but also can contribute to give more benefits to the actual converter. In my opinion the costs are not too high and the possible benefits will make the product more flexible and attractive for different trending sectors like for example smart energy monitoring.

Chapter 9

Conclusions and future works

9.1 Conclusions

The objectives fulfillments and subjects reviewed by this thesis are:

- A brief comparative between analog and digital control techniques has been done. It concludes that digital solution makes possible to add more functionalities and creates smarter power converters.
- Selected push-pull converter is being modeled by state space averaged model. During the first steps it was explored the possibility of including transformer non-ideal components like magnetization inductance in the model, finally it was rejected because it described only transient effects and it do not have sense with the model technique adopted.
- It have been applied voltage mode control design for the converter model and during that process it was proposed two different compensation algorithms, PID and PI+Lead. By simulation it was possible to test its steady state and transitory performance concluding that PI+Lead had faster response, while PID introduced less overshoot.
- It was studied which effects will bring in Pulse Width Modulation restrictions on designed controllers and it was observed that as compensation signal never reach duty cycle limits saturation, its restriction can be neglected. PWM block only introduced ripples for the output voltage and inductor current averaged values.
- For discrete-time control design, it was taken the proposed analog PID and PI+Lead controllers and discretized them by Tustin approach. Doing this, it was possible to prove that its performance was really close to analog designs under a certain sampling rate. This discarded the possibility of using discrete-time control design techniques because they were not needed.

- Sample frequency was selected to be equal to the converter switching frequency for avoid aliasing effects among other reasons explained in previous sections, giving a period time $T_s = 8[\mu s]$. It was tested that for values 2.5 times higher the sampled period selected, the system became unstable.
- During the thesis it was discovered the possibility of simulating the performance from digital controllers using PSIM and DLL files. This tool is very powerful because makes possible to check the code before connecting to hardware components, reducing possible damages on the prototype converter. With this tool PI+Lead controller was discarded by finally selecting PID.
- Converter digital control implementation is done with C2000 DSP from Texas Instruments, being fully accomplished the main objective of the thesis. It was possible to apply digital control to selected commercial power conversion stage.
- A set of experimental results were obtained from the final solution, load and line disturbances were really good and even better than manufacturer given specifications. Transient responses were close to the simulations, proving the designs. Despite this, ripple and noise waveforms obtained are worst than expected provably due to sensing noise that was not possible to eliminate by filtering.
- Costs estimations conclude that this solution implies an investment of 6.4% from the converter list price. With all the benefits that can be achieved with this solution it is considered a very good choice.
- Converter initial analog compensation expression was analyzed and studied by simulation, but as the capacitors have huge tolerances it was not possible to obtain exact poles and zeros. The experiment was not as interesting as expected by initial objectives.
- Thesis redaction is done using TexStudio and Latex based text editor, accomplishing initial objective.

9.2 Future works

Following the thesis studies, the proposed future works are:

- Converter output voltage ripple and noise was worst than expected, during the implementation it was expended a lot of efforts trying to solve this issue without successful results.

- To design and integrate all modifications in the converter Printed Circuit Board.
- During the design process, PI+Lead compensator was discarded because it was not possible to avoid steady state oscillations, try to solve this problems and implement the controller will improve converter dynamic response.
- One improvement line will be to give more functionalities to the converter using selected DSP. It was not the objective of this work but including some functionalities like communications is not difficult and will improve the specifications from the selected converter.
- The control loop selected is voltage mode, by applying Averaged Current Mode Control technique it will be possible to limit the inductance current peaks. The converter already senses the input current, which is proportional to the inductance one, then it should be good to design and implement an ACMC technique.
- Make extensible digital control application to the entire CRS-500 (selected converter) family.

Chapter 10

Annexes

10.1 Matlab files

Following pages include Matlab "*.m" files used for design the proposed PID and PI+Lead controllers. Designs done for continuous-time domain and its respective discretized compensation expressions are included.

CONVERTER PARAMETERS MATLAB FILE

File name: Converter_variables_CRS500_Real_values.m

```
%% INPUT VALUES CRS500 - 6465
close all; clear all; clc;

Vin = 110; % Input voltage (Volts) 77..144V
Vout = 48; % Output voltage (Volts) -10%..+15%
N1 = 11; % Primary transformer turns
N2 = 9; % Secondary transformer turns
D = (Vout*N1)/(2*Vin*N2); % Duty cycle

% Output Filter
L = 71.1*1e-6; % Inductance (Henries)
C = 6000*1e-6; % Output capacitor (Farads)
RL = 4.6; % Load Resistance (ohm): 6.09 - 3.73 ohm
W0 = 1/sqrt(L*C);
Q=RL*sqrt(C/L); % Quality factor

% UC3525 frequency parameters
Rt = 5111; % Switching resistor selector Rt
Rd = 4.7; % Switching resistor selector Rd
Ct = 2.2*1e-9; % Switching capacitance selector Ct
fosc = 1/(Ct*(0.7*Rt+3*Rd)); % Switching Frequency (Hz)
fs = 128*1e3; % Frequency (Hz)

Hs = 1; % It is 1 working with real V and I values

%Isense: Input current Transconductance
Rs = 1; % Current transformer Gain
Kc = 0.95; % Coupling coefficient
K = Kc*(N1/N2); % IL/Iin Ratio

Vm = 1; % PWM Modulator voltage [V]
PWM = 1/Vm; % PWM function
```

PI+LEAD VOLTAGE MODE CONTROL PM=60 ANALOG DESIGN FILE

File name: VMC_PM60_control_design_RV.m;

```
% ----- %
% VOLTAGE MODE CONTROL ANALYSIS %
% ----- %
% Author: Jaume Balcells Ortega
% Date: 11_09_2017
% Review: 30_01_2018
% version: 1.3

%% INPUT VALUES
Converter_variables_CRS500_Real_values;
```

```

fprintf('PROGRAM FOR EVALUATE THE VOLTAGE MODE CONTROL (VMC) TECHNIQUE OF
A PUSH-PULL CONVERTER:');
fprintf('\n');fprintf('\n');
fprintf('CRS500 PM = 60');
fprintf('\n');fprintf('\n');
P=bodeoptions;
P.FreqUnits='Hz';
P.Grid='on';

%% CONVERTER TRANSFER FUNCTIONS:
%CONTROL TO OUTPUT
Gvd = tf([Vin*N2/N1],[L*C L/RL 1]);

% ----- %
%% OPEN LOOP TRANSFER FUNCTION (GAIN ANALYSIS)

%OPENLOOP_GAIN = Gvd*Hsense*Gc*PWM ; with Gc=1
Gc = 1; % Control unitari gain
T = Gvd*Hs*Gc*PWM; % loop gain
%Open loop transfer function Bode plot
figure('name','T(s) Open loop transfer function with Gc=1');
% naming and title in the figures
bode(T);set(findall(gcf,'type','line'),'linewidth',2);
%Phase and gain margin
margin(T);grid; % [Gm,Pm,Wgm,Wpm] = margin(T)
set(findall(gcf,'type','line'),'linewidth',2);
% Transient response of Closed loop (Gc=1)
Tcl=T/(1+T);
figure;
step((Vout)*Tcl);grid;

% ----- %

%% PI CONTROL DESIGN:
% CONTROLLER EXPRESSION:
% Gc_pi = (Kp*s + Ki)/s

% GOALS:
% 1.- Steady state error null
% 2.- Phase Margin = 60°

%from the open loop bode (T) take Wc and G_wc values (analitical
analysis)
% Open-loop phase = -180 + PM_o + PM; being PM_o =10°and PM the target
margin.
Phase = -180 + 10 + 60; % real Phase = -100;
Wc = 1.54e3; % rad/s crossover frequency at desired PM
G_wc = 70.8; % dB at desired Wc phase for open loop
bode

Kp = 10^(-G_wc/20); % Kp = 0.026;
Ki = (Wc/10)*Kp; % Ki = 5.3823;
%Open loop transfer function
fprintf('PI CONTROLLER FdT:');
Gc_pi = tf([Kp Ki],[1 0]) % PI controller transfer function
G_pi = T * Gc_pi; % Plant + PI open loop transfer function
%Phase and gain margin

```



```

margin(G_pi);grid;set(findall(gcf,'type','line'),'linewidth',2);

%Open loop Bode plot: plant, PI controller and plant+controller(G_pi)
figure; % Bodes figure
bode(T,P);hold on; % Plant
bode(Gc_pi,P); % PI controller
bode(G_pi,P); % Plant+controller
set(findall(gcf,'type','line'),'linewidth',2);

%Nyquist Open loop transfer function of G_pi(s)
figure;
nyquist(G_pi);grid;set(findall(gcf,'type','line'),'linewidth',2);
% Closed loop Transient response with PI controller
G_cl_pi = feedback(G_pi,1); % Plant+PI control closed loop function
figure;

step((Vout)*G_cl_pi);grid;set(findall(gcf,'type','line'),'linewidth',2);

% ----- %

%% ZERO-POLE (LEAD) CONTROL DESIGN:
% CONTROLLER EXPRESSION:
%  $G_{c\_zp} = G_{c0} \frac{1+s/W_z}{1+s/W_p}$ 

% GOALS:
% 1.- Phase Margin =  $50^\circ$ 
% 2.- Crossover frequency  $fc2 = 3*fc$  (3times higher than the open
loop crossover frequency)

% Decide the crossover frequency fc2
% Analitical analysis for obtain phase G_wc2
[Gm,Pm,Wgm,Wpm] = margin(T); % Wc = Wpm
Wc = Wpm; % Obtain the real crossover frequency
after PI controller action
Wc2=3*Wc; % Wc2: Crossover frequency
G_Wc2 = -19.3; % [dB] Gain at the selected crossover
frequency (obtained analitically in the bode)
[Gm,Pm,Wgm,Wpm] = margin(G_pi);
PM=Pm; %PM=60; % Phase margin desired
% Pole compensator
A = (1-sin(deg2rad(PM)))/(1+sin(deg2rad(PM)));
Wp = Wc2/sqrt(A); %fp=Wp/2*pi;
% Zero compensator
Wz= A*Wp; %fz=Wz/2*pi;
% Compensator Gain
Gc0 = 10^((-G_Wc2 - 20*log10(sqrt(Wp/Wz)))/20);
% Controller transfer function:
fprintf('ZERO-POLE (LEAD) CONTROLLER FdT:');
Gc_zp = Gc0*tf([1/Wz 1],[1/Wp 1]) % Zero-Pole controller transfer
function
G_zp = T*Gc_zp; % Plant + Zero-Pole transfer function
%Phase and gain margin
margin(G_zp);grid;set(findall(gcf,'type','line'),'linewidth',2);
%Open loop Bode plot: plant(T), Zero-Pole controller(Gc_zp),
plant+controller(G_zp)
figure;
bode(T,P);hold on; % Plant

```

```

    bode(Gc_zp,P); % Zero-Pole (Lead) controller
    bode(G_zp,P); % Plant+controller
    set(findall(gcf,'type','line'),'linewidth',2);
%Nyquist Open loop transfer function of G_zp(s)
figure;
nyquist(G_zp);grid;set(findall(gcf,'type','line'),'linewidth',2);
% Closed loop Transient response with Zero-Pole controller
G_cl_zp = feedback(G_zp,1); % G_zp(s) closed loop function
figure;

step((Vout)*G_cl_zp);grid;set(findall(gcf,'type','line'),'linewidth',2);

% ----- %

%% PI + LEAD CONTROL DESIGN:
% CONTROLLER EXPRESSION:
% Gc_pi_zp = [Gc0*((1+s/Wz)/(1+s/Wp))] * [Wzpi*(1+s/Wzpi)/s]
% Gc_pi_zp = Gc0*Wzpi* [(1+s/Wz)*(1+s/Wzpi) / [s]*(1+s/Wp)]
% Gc_pi_zp = Gc_zp * Gc_pi_2
% GOALS:
% 1.- Steady state error null
% 2.- Phase Margin = 50°
% 3.- Crossover frequency fc2 = 3*fc (3times higher)

% DESIGN RULE: design Wzpi a decade above of the open loop resonance
Wzpi2 = 0.1*W0;
% Controller transfer function:
fprintf('PI + ZERO-POLE CONTROLLER FdT:');
Gc_pi_2 = Wzpi2*tf([1/Wzpi2 1],[1 0]); % PI controller transfer
function
Gc_pi_zp = Gc_pi_2*Gc_zp % PI + Zero-Pole controller
transfer function
G_pi_zp = T*Gc_pi_zp; % Plant + PI + Zero-Pole
transfer function
%Phase and gain margin
margin(G_pi_zp);grid;set(findall(gcf,'type','line'),'linewidth',2);
%Open loop Bode plot: plant(T), PI + Zero-Pole controller(Gc_pi_zp),
plant+controller(G_pi_zp)
figure;
bode(T,P);hold on; % Plant
bode(Gc_pi_zp,P); % PI + Zero-Pole controller
bode(G_pi_zp,P); % Plant+controller
set(findall(gcf,'type','line'),'linewidth',2);
%Nyquist Open loop transfer function of G_zp(s)
figure;
nyquist(G_pi_zp);grid;set(findall(gcf,'type','line'),'linewidth',2);
% Closed loop Transient response with Zero-Pole controller
G_cl_pi_zp = feedback(G_pi_zp,1); % G_pi_zp(s) closed loop
function
figure;

step((Vout)*G_cl_pi_zp);grid;set(findall(gcf,'type','line'),'linewidth',2);
);

close all;

```

PI+LEAD VMC PM=60 DISCRETIZATION DESIGN FILE

File name: VMC_PM60_Controller_discretization_RV.m;

```
% ----- %
%           CONTROLLER DISCRETITZATION           %
% ----- %
% Author: Jaume Balcells Ortega
% Date: 15_10_2017
% Review: 30_01_2018
% version: 1.0

close all;
clear all;
clc;
fprintf('PROGRAM FOR DISCRETIZE THE CONTROLLER FROM THE CONTINOUSE-TIME
DOMAIN DESIGN:');
fprintf('\n');fprintf('\n');
P=bodeoptions;
P.FreqUnits='Hz';
P.Grid='on';

VMC_PM60_control_design_RV; % Calling the Controller design .m file
close all; % Delete graphical figures
clc;
format long;
fprintf('CRS500 PM = 60°');
fprintf('\n');fprintf('\n');

% ----- %
%% CONTINOUSE TIME

Gc = Gc_pi_zp % Define the controller FdT
Gc_T = T*Gc; % Plant+controller FdT
G_cl_c = feedback(Gc_T,1); % Plant+controller feedback
step(Vout*G_cl_c);grid;set(findall(gcf,'type','line'),'linewidth',2);
% Step response closed loop
hold on;

% ----- %
%% DISCRETE TIME
Ts = 1/(fs); % Sampling time 1us (f = 124e03)
method = 'tustin'; % Bilinear tustin aproximation method

Gc_d = c2d(Gc,Ts,method) % Discretized controller function
T_d = c2d(T,Ts,method); % Discretized Plant function
Gc_d_T = T_d*Gc_d; % Plant+controller FdT
G_cl_c_d = feedback(Gc_d_T,1); % Plant+controller feedback
step(Vout*G_cl_c_d);grid;set(findall(gcf,'type','line'),'linewidth',2);
% Step response closed loop
close all;
```

PID VOLTAGE MODE CONTROL PM=60 ANALOG AND DIGITAL DESIGN FILE

File name: VMC_PID_PM60.m;

```
% ----- %
%          VOLTAGE MODE CONTROL ANALYSIS          %
% ----- %
% Author: Jaume Balcells Ortega
% Date: 11_09_2017
% Review: 29_01_2018
% version: 1.2

%% INPUT VALUES
Converter_variables_CRS500_Real_values;

fprintf('PROGRAM FOR EVALUATE THE VOLTAGE MODE CONTROL (VMC) TECHNIQUE OF
A PUSH-PULL CONVERTER:');
fprintf('\n');fprintf('\n');
fprintf('PID - CRS500 PM = 60');
fprintf('\n');fprintf('\n');
P=bodeoptions;
P.FreqUnits='Hz';
P.Grid='on';

%% CONVERTER TRANSFER FUNCTIONS
%CONTROL TO OUTPUT
Gvd = tf([Vin*N2/N1],[L*C L/RL 1]);

% ----- %
% Automatic PID tuning
opts = pidtuneOptions('PhaseMargin',60);
[C1, info] = pidtune(Gvd, 'pid', opts)

% - CONTINUOUS TIME ----- %
G_pid = Gvd*C1;
G_cl_pid = feedback(G_pid,1);
figure;
step((Vout)*G_cl_pid);grid;set(findall(gcf,'type','line'),'linewidth',2)
hold on

% - DISCRETE TIME ----- %
Ts = 1/(fs); %1e-06; % Sampling time 1us (f = 124e03)
method = 'Trapezoidal'; % Bilinear tustin approximation method

PID_disc = pid(C1.Kp,C1.Ki,C1.Kd,0,1/fs,'IFormula',method)

Gvd_disc = c2d(Gvd,Ts,method); % Discretized Plant function

GcPID1_d_T = Gvd_disc*PID_disc; % Plant+PID controller FdT
G_cPID1_d_cl = feedback(GcPID1_d_T,1); % Plant+PID controller feedback

step(Vout*G_cPID1_d_cl);grid;set(findall(gcf,'type','line'),'linewidth',2)
); % Step response closed loop
close all;
```

10.2 PSIM controllers DLLs

To simulate the behavior of discrete-time controllers in PSIM it was developed Dynamic Link Library files using visual studio. It is included the DLLs codes used for the PID and PI+Lead simulations.

PI+LEAD CONTROLLER PSIM DLL FILE

File name: 2P2Z_controller_inp.cpp

```
/* ----- */
/*          PI+LEAD CONTROLLER PSIM DLL          */
/* ----- */
/* Author: Jaume Balcells Ortega                  */
/* Date: 02_03_2018                             */

/* This code implements a discrete time PI+Lead control law where
parameters coefficients are DLL bloc inputs */

/*CONTROL LAW*/
//  $u(k) = b_0 \cdot e(k) + b_1 \cdot e(k-1) + b_2 \cdot e(k-2) - a_1 \cdot u(k-1) - a_2 \cdot u(k-2)$ 

/*INPUTS*/
//  $u(k) = in[0]$ ; : controller input signal
//  $b_0 = in[1]$ ; : controller input coefficient
//  $b_1 = in[2]$ ; : controller input coefficient
//  $b_2 = in[3]$ ; : controller input coefficient
//  $a_1 = in[4]$ ; : controller input coefficient
//  $a_2 = in[5]$ ; : controller input coefficient

/*OUTPUTS*/
//  $e(k) = out[0]$ ; : controller output signal

#include <math.h>
#include <stdio.h>

/*CONTROLLER INPUTS*/
static double Voutk = 0.0; // Vout[k]
static double Voutk1 = 0.0; // Vout[k-1]
static double Voutk2 = 0.0; // Vout[k-2]
/*CONTROLLER OUTPUTS*/
static double Vcontk = 0.0; // Vcontrol[k]
static double Vcontk1 = 0.0; // Vcontrol[k-1]
static double Vcontk2 = 0.0; // Vcontrol[k-2]

__declspec(dllexport) void simuser(double t, double delt, double *in,
double *out)
{
    Voutk = in[0]; // read DLL input Vout[k]

    Vcontk = (in[1] * Voutk) +
        (in[2] * Voutk1) +
        (in[3] * Voutk2) -
        (in[4] * Vcontk1) -
        (in[5] * Vcontk2); // calculate the CONTROL LAW

    Vcontk2 = Vcontk1; // Vcontrol[k-2] = Vcontrol[k-1]
    Vcontk1 = Vcontk; // Vcontrol[k-1] = Vcontrol[k]
    Voutk2 = Voutk1; // Vout[k-2] = Vcontrol[k-1]
    Voutk1 = Voutk; // Vout[k-1] = Vcontrol[k]
    out[0] = Vcontk; // Write DLL output Vcontrol[k]
}
```

PID CONTROLLER PSIM DLL FILE

File name: PID_controller.cpp

```
/* ----- */
/*          PID CONTROLLER PSIM DLL          */
/* ----- */
/* Author: Jaume Balcells Ortega             */
/* Date: 02_03_2018                         */

/* This code implements a discrete time PID control law where parameters
coefficients are DLL bloc inputs */
#include <math.h>
#include <stdio.h>

/*CONTROL LAW*/
//  $u(k) = A \cdot e(k) + B \cdot e(k-1) + C \cdot e(k-2) + u(k-1)$ 
//  $A = K_p + K_i \cdot (T_s/2) + K_d \cdot (1/T_s);$  // A coefficient
//  $B = -K_p + K_i \cdot (T_s/2) - 2 \cdot K_d \cdot (1/T_s);$  // B coefficient
//  $C = K_d \cdot (1 / T_s);$  // C coefficient

/*INPUTS*/
//  $u(k) = in[0];$  : controller input signal
//  $K_p = in[1];$  : controller input Proportional Gain coefficient
//  $K_i = in[2];$  : controller input Integral Gain coefficient
//  $K_d = in[3];$  : controller input Derivative Gain coefficient
//  $T_s = in[4];$  : controller input sampling period [s]
//  $A = (in[1] + (in[2] \cdot (T_s / 2)) + (in[3] \cdot (1 / T_s)));$ 
//  $B = (-in[1] + (in[2] \cdot (T_s / 2)) - (2 \cdot in[3] \cdot (1 / T_s)));$ 
//  $C = (in[3] \cdot (1 / T_s));$ 

/*OUTPUTS*/
//  $e(k) = out[0];$  : controller output signal

/*CONTROLLER INPUTS*/
static double Voutk = 0.0; // Vout[k] = e[k]
static double Voutk1 = 0.0; // Vout[k-1] = e[k-1]
static double Voutk2 = 0.0; // Vout[k-2] = e[k-2]
/*CONTROLLER OUTPUTS*/
static double Vcontk = 0.0; // Vcontrol[k] = u[k]
static double Vcontk1 = 0.0; // Vcontrol[k-1] = u[k-1]

__declspec(dllexport) void simuser(double t, double delt, double *in,
double *out)
{
    Voutk = in[0]; // read DLL input Vout[k]
    // calculate the CONTROL LAW
    Vcontk = ((in[1] + (in[2] * (in[4] / 2)) + (in[3] * (1 / in[4])) * Voutk) +
              ((-in[1] + (in[2] * (in[4] / 2)) -
              (2 * in[3] * (1 / in[4])) * Voutk1) +
              ((in[3] * (1 / in[4])) * Voutk2) + (Vcontk1);
    Vcontk1 = Vcontk; Voutk2 = Voutk1; Voutk1 = Voutk;
    out[0] = Vcontk; // Write DLL output Vcontrol[k]
}
```

10.3 Firmware code

The DSP controller firmware code used to obtain the experimental results is included in the following pages:


```

// *****
// *****
// FILE:   PushPull_VMC_JBO_PID_v7.c
// Author: Jaume Balcells Ortega
// Description: VMC-PID in a push-pull converter
// Date: 02_03_18
// *****
// *****

#include "F28x_Project.h" // Device Header File and Examples Include File
#include "DCL.h"          // Digital controller library
#include "math.h"         // Mathematical operations library

/*-----*/
/*          CONVERTER PARAMETERS          */
/*-----*/
float Vout_nom = 48.0f; // [V] nominal output voltage
float Vout_min = 0.0f; // [V] minimum output voltage
float Vout_min_alarm = 42.0f; // [V] minimum output voltage alarm
float Vout_max = 55.2f; // [V] maximum output voltage
float Vout_max_alarm = 54.0f; // [V] maximum output voltage alarm
float Vin_nom = 110.0f; // [V] nominal input voltage
float Vin_min = 80.0f; // [V] minimum output voltage
float Vin_max = 145.0f; // [V] maximum output voltage
float Vin_max_alarm = 140.0f; // [V] maximum output voltage alarm
float Iin_nom = 4.54f; // [A] nominal input current
float Iin_min = 0.0f; // [A] minimum input current
float Iin_max = 6.5f; // [A] maximum input current
float Iin_max_alarm = 6.0f; // [A] maximum input current alarm
int N1 = 11; // Transformer Primary winding turns
int N2 = 9; // Transformer Secondary winding turns

/*-----*/
/*          PULSE WIDTH MODULATION          */
/*-----*/
#define PWM_PERIOD 0x030D // PWM period= 781counts= 64kHz(1count=10ns)
#define PWM_INIT25 (PWM_PERIOD>>2) // PWM initial duty cycle = 25% = T/4
#define PWM_INIT0 0x0000 // PWM initial duty cycle = 0%
#define PWMAB_OFFSET (PWM_PERIOD>>1) // PWM offset between A and B is T/2

Uint16 period = PWM_PERIOD; // PWM period (n# counts) variable
Uint16 dutyCycleA = PWM_INIT0; // PWM1 duty cycle A = 0%
Uint16 dutyCycleB = PWM_INIT0; // PWM5 duty cycle B = 0%
Uint16 pwmABoffset = PWMAB_OFFSET; // PWMA-PWMB phase offset = 180°
float DC = 0.0f; // [%] DC Overall duty PWMA and PWMB
Uint16 PWM = 0; // PWM DC adapted to clock counts
Uint16 DCmin = 1; // [%] DC minimum
Uint16 DCmax = 45; // [%] DC maximum

/*-----*/
/*          VOUT SENSE          */
/*-----*/
float ADC12FS = 4096.0f; // 12 bits Full Scale ADC value
float Vcc = 3.3f; // [V] ADC supply voltage
float Vout_aten = 20.45f; // [V/V] Vout att. = 20.45 (after calib.)
float Vsoftstart = 0.0f; // [V] soft-start reference output voltage
float Vout_ok = 0.0f; // [V] Output volt superv. after soft-start

```

```

/*-----*/
/*          VIN SENSE          */
/*-----*/
float Vin = 0.0f;           // [V] Input voltage superv. after soft-start
float Vin_aten = 43.93f; // [V/V] Vin sense attenuation = 43.93

/*-----*/
/*          IIN SENSE          */
/*-----*/
float Iin = 0.0f;           // [A] Input current supervisor after soft-start
float Iin_aten = 1.96f; // [A/V] Iin sense attenuation = 1.96

/*-----*/
/*          VOUT PID CONTROLLER      */
/*-----*/
float Kp = 0.0f, Ki = 0.0f, Kd = 0.0f; // PID coef. initialization
float A = 0.0f, B = 0.0f, C = 0.0f; // PID coef. initialization
float ek = 0.0f, ek1 = 0.0f, ek2 = 0.0f; // e[k]: error signal
(controller input)
float uk = 0.0f, uk1 = 0.0f, uk2 = 0.0f; // u[k]: control signal
(controller output)
float Ts = 7.8125e-6f; // Ts : Controller sampling period (Tustin)

float Vs = 0.0f; // [V] Output voltage sample
float Vs_offset = 0.235f; // [V] Output voltage Offset //0.08

/*-----*/
/*          SOFT-START          */
/*-----*/
int ramp_time = 10000; // 10.000[ms] Soft_start_time

/*-----*/
/*          FUNCTION DECLARATION      */
/*-----*/
void Setup_ADC(void); // ADC: Init+Config - ADCINA3 (pin #26)
void Setup_ADCsample(void); // ADC-Sampling Trigger: Init+Config
void Init_ADCsample(void); // PWM-ADC: Configure ePWM module 2
void Init_PWMA(void); // PWM-A: Config GATE-A PWM - PWM1A(pin #40)
void Init_PWMB(void); // PWM-B: Config GATE-B PWM - PWM5A(pin #78)
void Soft_Start(int ms_ramp_time); // Soft-Start delay
void Error_checker(void); // Routine for check if an error happened

interrupt void controller_isr(void); // Control interrupt service routine

void main(void)
{
/*-----*/
/*          CONTROLLER COEFFICIENTS      */
/*-----*/

//PID with PM = 60
Kp = 4.1469e-2f;
Ki = 3.114029327267692f;
Kd = 1.605457967637553e-05f;

```

```

A = Kp + Ki*(Ts/2) + Kd*(1/Ts);    // A coefficient
B = -Kp + Ki*(Ts/2) - 2*Kd*(1/Ts); // B coefficient
C = Kd*(1 / Ts);                  // C coefficient

/*-----*/
/*      INITIALIZATIONS          */
/*-----*/
// Initialize System Control
InitSysCtrl();
EALLOW;
ClkCfgRegs.PERCLKDIVSEL.bit.EPWMCLKDIV = 1;
EDIS;           // Initialize GPIO
InitGpio();     // Configure default GPIO
InitEPwm1Gpio(); // Configure EPWM1 GPIO pins
InitEPwm5Gpio(); // Configure EPWM5 GPIO pins
EALLOW;
GpioCtrlRegs.GPADIR.bit.GPIO31 = 1; // Drives LD2 on controlCARD
GpioCtrlRegs.GPADIR.bit.GPIO19 = 1; // GPIO19 (pin3) contr. ISR
EDIS;
GpioDataRegs.GPADAT.bit.GPIO31 = 1; // Turn off LED
GpioDataRegs.GPADAT.bit.GPIO19 = 0; // Turn off GPIO19
// Clear all interrupts and initialize PIE vector table
DINT;
InitPieCtrl();
IER = 0x0000;
IFR = 0x0000;
InitPieVectTable();
// Map ISR functions
EALLOW;
PieVectTable.ADCA1_INT = &controller_isr; // ADCA interrupt 1
EDIS;
// ADC
Setup_ADC(); // ADC: Setup and Power-up
Setup_ADCsample(); // ADC-sampling trigger Setup ADCINA3
// PWM
Init_PWMA();
Init_PWMB();
Init_ADCsample();

/*-----*/
/*      START-UP                */
/*-----*/
Soft_Start(ramp_time); // [ms] Soft Start sequence

Vout_ok = ((AdcaResultRegs.ADCRESULT0)*((Vcc/ADC12FS))*Vout_aten);

if ((Vout_ok >= Vout_max_alarm) && (Vout_ok <= Vout_min_alarm))
{
    DC = 0.0f; PWM = DC*period; // Stop DC
    /*ACTUALIZE PWMA and PWMB*/
    EPwm1Regs.CMPA.bit.CMPA = PWM; //dutyCycleA;
    EPwm5Regs.CMPA.bit.CMPA = PWM; //dutyCycleB

    Soft_Start(ramp_time);

    float Vout_ok2;

```

```

Vout_ok2 =
((AdcaResultRegs.ADCRESULT0)*((Vcc/ADC12FS))*Vout_aten);

// double sensing error?
if ((Vout_ok2 >= Vout_max ) && ( Vout_ok2 <= Vout_min ))
{
    DC = 0.0f; PWM = DC*period; // Stop DC
    EPwm1Regs.CMPA.bit.CMPA = PWM;//dutyCycleA;
    EPwm5Regs.CMPA.bit.CMPA = PWM;//dutyCycleB
    Vout_ok2 = 0.0f;
    Error_checker(); // call error check function
}

else
{
    // Enable global interrupts
    IER |= M_INT1; // Enable group 1 interrupts
    EINT; // Enable Global interrupt INTM
    ERTM; // Enable Global realtime interrupt DBGM
    // Enable PIE interrupt
    PieCtrlRegs.PIEIER1.bit.INTx1 = 1;
    // Sync ePWM
    EALLOW;
    CpuSysRegs.PCLKCR0.bit.TBCLKSYNC = 1;
    // Start ePWM
    EPwm2Regs.ETSEL.bit.SOCAEN = 1; // Enable SOCA
    EPwm2Regs.TBCTL.bit.CTRMODE = 0; // Un-freeze&enter up-count mode
    // Blinking LEDs Routine
    do {
        GpioDataRegs.GPADAT.bit.GPIO31 = 0; // Turn on LED
        DELAY_US(1000 * 500); // ON delay
        GpioDataRegs.GPADAT.bit.GPIO31 = 1; // Turn off LED
        DELAY_US(1000 * 500); // OFF delay
        Error_checker(); // Check errors
    } while(1);
}

}

interrupt void controller_isr(void)
{
    /*ADC ACQUISITION*/
    Vs = ((AdcaResultRegs.ADCRESULT0)*((Vcc/ADC12FS))*Vout_aten);
    Vs = Vs - (Vs_offset*Vout_aten);
    if (Vs <= 0.0f)
        Vs = 0.0f;
    if ((Vs >= Vout_max ) && ( Vs <= Vout_min ))
        {Error_checker();}

    /*CONTROL LAW CALCULATION*/
    // u(k) = A*e(k) + B*e(k-1) + C*e(k-2) + u(k-1)
    ek = Vout_nom - Vs;
    uk = A*ek + B*ek1 + C*ek2 + uk1;
    uk1 = uk;
    ek2 = ek1;
}

```

```

    ek1 = ek;

    DC = uk;
    PWM = DC*period;

    if (PWM >= (DCmax*period/100) )    // Positive Saturation
        {PWM = DCmax*period/100;}
    else if (PWM <= (DCmin*period/100)) // Negative Saturation :
        {PWM = DCmin;}

    /*ACTUALIZE PWMA and PWMB*/
    EPwm1Regs.CMPA.bit.CMPA = PWM;    //dutyCycleA;
    EPwm5Regs.CMPA.bit.CMPA = PWM;    //dutyCycleB

    // Return from interrupt
    AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;    // Clear ADC INT1 flag
    PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;    // Acknowledge PIE group 1
to enable further interrupts
}

// Write ADC config and power up the ADC for ADC A, B and C Regs
void Setup_ADC(void)
{
    EALLOW;
    // ADC-A: OUTPUT VOLTAGE
    AdcaRegs.ADCCTL2.bit.PRESCALE = 6;    // Set ADCCLK divider to /4
    AdcaRegs.ADCCTL2.bit.RESOLUTION = 0;    // 12-bit resolution
    AdcaRegs.ADCCTL2.bit.SIGNALMODE = 0;    // Single-ended channel
conversions (12-bit mode only)
    AdcaRegs.ADCCTL1.bit.INTPULSEPOS = 1;    // Set pulse positions late
    AdcaRegs.ADCCTL1.bit.ADCPWDNZ = 1;    // Power up the ADC
    EDIS;
    // ADC-B: INPUT VOLTAGE
    AdcbRegs.ADCCTL2.bit.PRESCALE = 6;
    AdcbRegs.ADCCTL2.bit.RESOLUTION = 0;
    AdcbRegs.ADCCTL2.bit.SIGNALMODE = 0;
    AdcbRegs.ADCCTL1.bit.INTPULSEPOS = 1;
    AdcbRegs.ADCCTL1.bit.ADCPWDNZ = 1;
    // ADC-C: INPUT CURRENT
    AdccRegs.ADCCTL2.bit.PRESCALE = 6;
    AdccRegs.ADCCTL2.bit.RESOLUTION = 0;
    AdccRegs.ADCCTL2.bit.SIGNALMODE = 0;
    AdccRegs.ADCCTL1.bit.INTPULSEPOS = 1;
    AdccRegs.ADCCTL1.bit.ADCPWDNZ = 1;
    EDIS;

    DELAY_US(1000); // Delay for 1ms to allow ADC time to power up
}

// PWM-ADC sampling trigger SETUP
void Setup_ADCsample(void)
{
    // Select the channels to convert and end of conversion flag
    EALLOW;
    AdcaRegs.ADCSOC0CTL.bit.CHSEL = 3;    // SOC0 will convert pin A3
    AdcaRegs.ADCSOC0CTL.bit.ACQPS = 14;    // Sample window is 100 SYSCLK
    AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 7; // Trigger on ePWM2 SOCA/C

```

```

        AdcaRegs.ADCINTSEL1N2.bit.INT1SEL = 0; // End of SOC0 set INT1 flag
        AdcaRegs.ADCINTSEL1N2.bit.INT1E = 1;  // Enable INT1 flag
        AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1; // Clear INT1 flag
        EDIS;
    }

// GATE-A - PWM-1A (launchpad pin40)
void Init_PWMA(void)
{
    // Setup TBCLK
    EPwm1Regs.TBCTL.bit.CTRMODE = 0; // Count up
    EPwm1Regs.TBPRD = period;        // Set timer period
    EPwm1Regs.TBCTL.bit.PHSEN = 0;   // Disable phase loading
    EPwm1Regs.TBPHS.bit.TBPHS = 0x0000; // Phase is 0
    EPwm1Regs.TBCTR = 0x0000;        // Clear counter
    EPwm1Regs.TBCTL.bit.HSPCLKDIV = 1; // Clock ratio to SYSCLKOUT
    EPwm1Regs.TBCTL.bit.CLKDIV = 0;
    EPwm1Regs.TBCTL.bit.SYNCSEL = 1; // SYNC output on CTR = 0
    // Setup shadow register load on ZERO
    EPwm1Regs.CMPCTL.bit.SHDWAMODE = 0;
    EPwm1Regs.CMPCTL.bit.SHDWBMODE = 0;
    EPwm1Regs.CMPCTL.bit.LOADAMODE = 0;
    EPwm1Regs.CMPCTL.bit.LOADBMODE = 0;
    // Set Compare values
    EPwm1Regs.CMPA.bit.CMPA = dutyCycleA; // Set compare A value
    // Set actions
    EPwm1Regs.AQCTLA.bit.ZRO = 2; // Set PWM1A on Zero
    EPwm1Regs.AQCTLA.bit.CAU = 1; // Clear PWM1A on event A, up count
}

// GATE-B - PWM-5A (launchpad pin78)
void Init_PWMB(void)
{
    // Setup TBCLK
    EPwm5Regs.TBCTL.bit.CTRMODE = 0; // Count up
    EPwm5Regs.TBPRD = PWM_PERIOD;    // Same period as PWM1
    EPwm5Regs.TBCTL.bit.PHSEN = 1;   // Enable phase loading
    EPwm5Regs.TBPHS.bit.TBPHS = pwmABoffset; // PWMA-PWMB Phase shift
    EPwm5Regs.TBCTR = 0x0000;        // Clear counter
    EPwm5Regs.TBCTL.bit.HSPCLKDIV = 1; // Clock ratio SYSCLKOUT
    EPwm5Regs.TBCTL.bit.CLKDIV = 0;
    // Setup shadow register load on ZERO
    EPwm5Regs.CMPCTL.bit.SHDWAMODE = 0;
    EPwm5Regs.CMPCTL.bit.SHDWBMODE = 0;
    EPwm5Regs.CMPCTL.bit.LOADAMODE = 0;
    EPwm5Regs.CMPCTL.bit.LOADBMODE = 0;
    // Set Compare values
    EPwm5Regs.CMPA.bit.CMPA = dutyCycleB; // Set compare A value
    // Set actions
    EPwm5Regs.AQCTLA.bit.ZRO = 2; // Set PWM1A on Zero
    EPwm5Regs.AQCTLA.bit.CAU = 1; // Clear PWM1A on event A, up count
}

```

```

// PWM-ADC sampling trigger INITIALIZATION
void Init_ADCsample(void)
{
    EALLOW;
    // Assumes ePWM clock is already enabled
    EPwm2Regs.TBCTL.bit.CTRMODE = 3;    // Freeze counter
    EPwm2Regs.TBCTL.bit.HSPCLKDIV = 0;   // TBCLK pre-scaler = /1
    EPwm2Regs.TBPRD = 0x61A;            // sampling = 1562counts (64kHz)
    EPwm2Regs.ETSEL.bit.SOCAEN = 0;     // Disable SOC on A group
    EPwm2Regs.ETSEL.bit.SOCASEL = 2;    // Select SOCA on period match
    EPwm2Regs.ETSEL.bit.SOCAEN = 1;     // Enable SOCA
    EPwm2Regs.ETPS.bit.SOCAPRD = 1;     // Generate pulse on 1st event
    EDIS;
}

// Gets [ms]time and makes a Ramp-up Soft-Start sequence
void Soft_Start(int ms_ramp_time)
{
    int startup_time = ms_ramp_time;    // [us] Soft Start time
    int SoftStart_ramp_counter;         // Ramp cycles counter
    int SoftStart_ramp_cycles=100;      // Ramp cycles number
    float SoftStart_ramp_step = 0.2f;   // [V] Ramp step increments

    for (SoftStart_ramp_counter= 0; SoftStart_ramp_counter <
        SoftStart_ramp_cycles; SoftStart_ramp_counter++)
    {
        Vsoftstart += SoftStart_ramp_step; // Add step to Vref

        /*DUTY CYCLE CALCULATION*/
        DC = ((Vsoftstart*N1)/(2*Vin_nom*N2));

        /*PWM #counts ADAPTION*/
        PWM = DC*period;

        /*MAXIMUM DUTY CYCLE CORRECTIONS*/
        if (PWM > (DCmax*period/100) )
        {PWM = DCmax*period/100;}
        else if (PWM <= (DCmin*period/100))
        {PWM = DCmin;}

        /*ACTUALIZE PWMA and PWMB*/
        EPwm1Regs.CMPA.bit.CMPA = PWM;    //dutyCycleA;
        EPwm5Regs.CMPA.bit.CMPA = PWM;    //dutyCycleB

        /*DELAY FOR ASSURE START-UP TIME*/
        DELAY_US(1000* (startup_time/SoftStart_ramp_cycles)); // [us]
    }
}

```

```

// Measures Vin and Iin and STOP CONVERTER IN CASE OF FATAL ERROR
function
void Error_checker(void)
{
    float Vin_ok;
    float Iin_ok;
    Vin_ok = ((AdcbResultRegs.ADCRESULT0)*((Vcc/ADC12FS))*Vin_aten);
    // Vin sense
    Iin_ok = ((AdccResultRegs.ADCRESULT0)*((Vcc/ADC12FS))*Iin_aten);
    // Iin sense

    // Iin or Vin upper limits?
    if ((Vin_ok >= Vin_max_alarm) || (Iin_ok >= Iin_max_alarm))
    {
        EPwm1Regs.CMPA.bit.CMPA = 0;           //dutyCycleA
        EPwm5Regs.CMPA.bit.CMPA = 0;           //dutyCycleB
        do {
            // Turn on FATAL ERROR LED
            GpioDataRegs.GPADAT.bit.GPIO31 = 0;
        } while(1);
    }

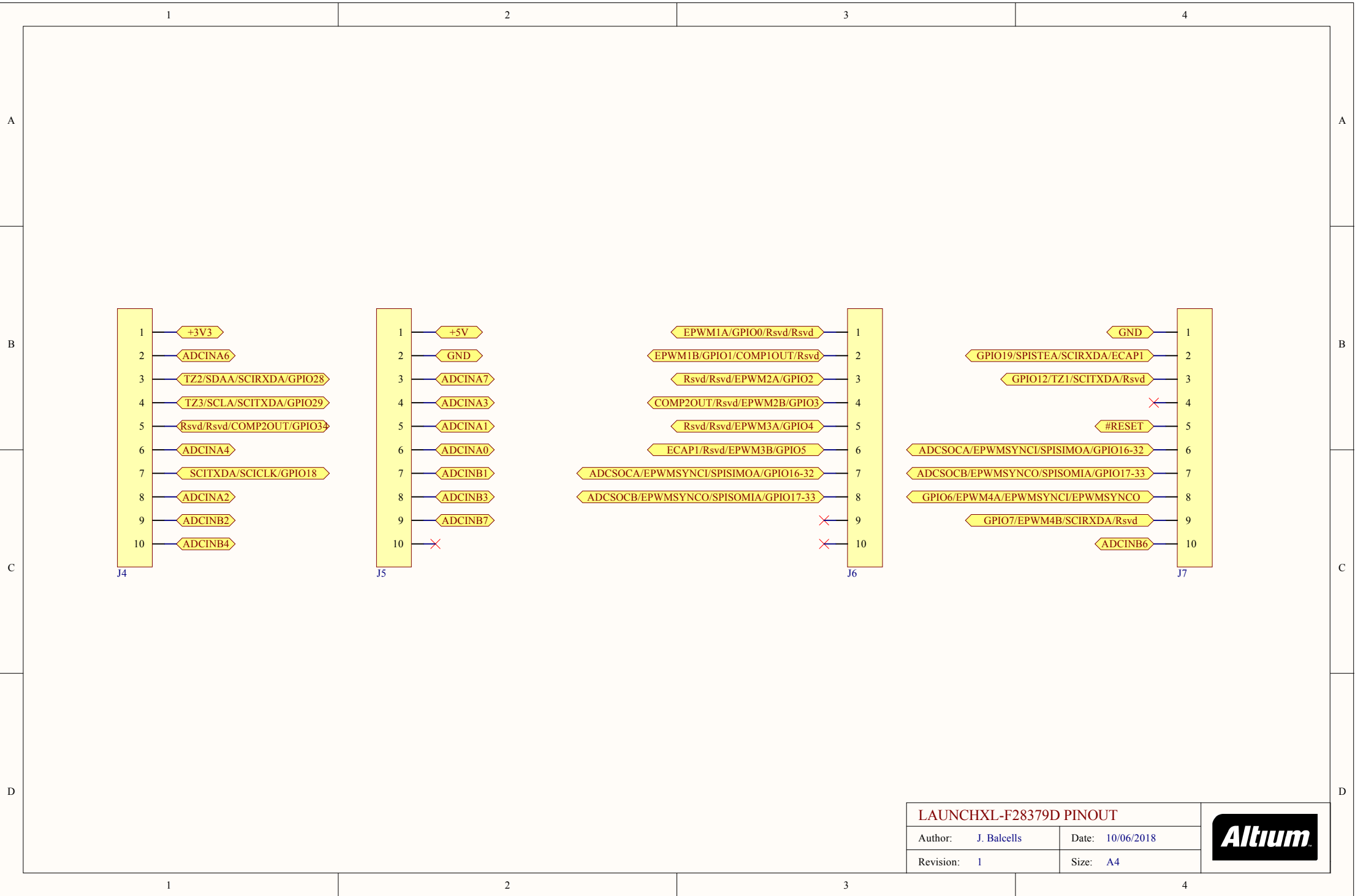
    else
    {
        Vin_ok = 0.0f;
        Iin_ok = 0.0f;
        Soft_Start(ramp_time);                 //[ms] Soft Start sequence
    }
}

//END OF FILE

```


10.4 Hardware Schematic

Hardware modifications schematic is included in the following pages.



J5

1

EPWM1A/GPIO0/Rsvd/Rsvd

2

EPWM1B/GPIO1/COMP1OUT/Rsvd

3

Rsvd/Rsvd/EPWM2A/GPIO2

4

COMP2OUT/Rsvd/EPWM2B/GPIO3

5

Rsvd/Rsvd/EPWM3A/GPIO4

6

ECAP1/Rsvd/EPWM3B/GPIO5

7

ADCSOCA/EPWMSYNCI/SPISIMOA/GPIO16-32

8

ADCSOCB/EPWMSYNCO/SPISOMIA/GPIO17-33

9

10

J6

1

GND

2

GPIO19/SPISTEA/SCIRXDA/ECAP1

3

GPIO12/TZ1/SCITXDA/Rsvd

4

5

#RESET

6

ADCSOCA/EPWMSYNCI/SPISIMOA/GPIO16-32

7

ADCSOCB/EPWMSYNCO/SPISOMIA/GPIO17-33

8

GPIO6/EPWM4A/EPWMSYNCI/EPWMSYNCO

9

GPIO7/EPWM4B/SCIRXDA/Rsvd

10

ADCINB6

J7

LAUNCHXL-F28379D PINOUT		
Author: J. Balcells	Date: 10/06/2018	
Revision: 1	Size: A4	

Bibliography

- [1] Luca Corradini, Dragan Maksimović, Paolo Mattavelli and Regan Zane. *Digital Control of High-Frequency Switched-Mode Power Converters*. 1st ed. Wiley, 2015.
- [2] R.W.Erickson and Dragan Maksimović. *Fundamentals of Power Electronics*. 2nd ed. Springer, 2001.
- [3] Francesc Guinjoan Gispert and Domingo Biel Solé. *Power Control and Processing subject class-notes*. UPC-ETSETB, 2017.
- [4] Katsuhiko Ogata. *Modern control engineering*. 5th ed. Pearson, 2010.
- [5] Ned Mohan, Tore M. Undeland, William P. Robbins. *Power Electronics: Converters, Applications and Design*. 3rd ed. Wiley, 2002.
- [6] M. Sami Fadali, Antonio Visioli. *Digital Control Engineering - Analysis and Design*. 2nd ed. Elsevier, 2013.
- [7] Krzysztof Sozański. *Digital Signal Processing in Power Electronics Control Circuits*. 1st ed. Springer, 2013.
- [8] Christophe Batard, Frédéric Poitiers, Christophe Millet and Nicolas Ginot. *Simulation of Power Converters Using Matlab-Simulink..*
<http://dx.doi.org/10.5772/46419>
- [9] Laszlo Balogh. *A Practical Introduction to Digital Power Supply Control*. Texas instrument SEM1600 Topic 6, 2005.
<http://www.ti.com/lit/ml/slup232/slup232.pdf>
- [10] P.R.K Chetty. *Modeling and deisgn of switching regulators*. IEEE transactions on aerospace and electric systems. vol. AES-18, n.3 , may 1982.
- [11] Guohua Zhou, Jianping Xu and Jinping Wang. *Constant-frequency peak-ripple-based control of buck converter in CCM: Review, unification and duality*. IEEE transactions on industrial electronics. vol. 61, n.3 , march 2014.
- [12] Finn Haugen. *Discrete-time signals and systems*. TechTeach, february 2005.
<http://techteach.no>
- [13] C2000 Digital Controller Library User's Guide. *SPRUI31*.
www.ti.com/cn/cn/lit/ug/sprui31/sprui31.pdf

- [14] TMS320F2837xD Dual-Core Delfino Microcontrollers Technical Reference Manual. *SPRS880H*.
<http://www.ti.com/lit/ug/spruhm8g/spruhm8g.pdf>
- [15] CRS-500 Premium converter Technical Reference Manual. *CRS500 datasheet*.
http://www.premium.es/web/upload/productos/24_es0241.pdf
- [16] DCPS Programmable DC Power Supply Technical Reference Manual. *Cinergia DCPS datasheet*. http://www.cinergia.coop/sites/default/files/fitxadcps_v2.pdf
- [17] DESIGNCORE: Start/Stop boost converter reference design. *D3 engineering fact sheet*.
<https://www.d3engineering.com/assets/uploads/data-sheets/connected-automation/D3Eng-DesignCore-StartStopBoost-RefDesign-FactSheet.pdf>
- [18] Basic calculation of a Buck Converter's Power Stage. *Richtek AN041 Application note*.
[https://www.richtek.com/en/Design\%20Support/Technical\%20Document/~media/AN\%20PDF/AN041_EN.ashx](https://www.richtek.com/en/Design\%20Support/Technical\%20Document/~/media/AN\%20PDF/AN041_EN.ashx)
- [19] LL4148-GS08 Technical Reference Manual. *LL4148-GS08 datasheet*.
<https://www.mouser.es/datasheet/2/427/114148-105055.pdf>
- [20] BYG10M-E3/TR Technical Reference Manual. *BYG10M-E3/TR datasheet*.
<https://www.mouser.es/datasheet/2/427/byg10-1147467.pdf>
- [21] NMV0505SAC Technical Reference Manual. *NMV0505SAC datasheet*.
<https://docs-emea.rs-online.com/webdocs/162e/0900766b8162e1b6.pdf>
- [22] BZX84C15LT1 Technical Reference Manual. *BZX84C15LT1 datasheet*.
<https://www.mouser.es/datasheet/2/308/bzx84c2v41t1-d-1190606.pdf>
- [23] MF-MSMF050-2 Technical Reference Manual. *MF-MSMF050-2 datasheet*.
<http://www.bourns.com/docs/Product-Datasheets/mfmsmf.pdf>
- [24] LMC6482AIM Technical Reference Manual. *LMC6482AIM datasheet*.
<http://www.ti.com/lit/ds/symlink/lmc6482.pdf>
- [25] UCC37324 Technical Reference Manual. *UCC37324 datasheet*.
<http://www.ti.com/lit/ds/symlink/ucc27324.pdf>
- [26] TL431CPL Technical Reference Manual. *TL431CPL datasheet*.
<http://www.ti.com/lit/ds/symlink/tl432b.pdf>
- [27] LM1117 Technical Reference Manual. *LM1117 datasheet*.
<http://www.ti.com/lit/ds/symlink/lm1117.pdf>
- [28] HCPL7800 Technical Reference Manual. *HCPL7800 datasheet*.
<https://docs.broadcom.com/docs/AV02-0410EN>
- [29] IRFI740G Technical Reference Manual. *IRFI740G datasheet*.
<https://www.vishay.com/docs/91156/sihfi740.pdf>

- [30] BUT11AF Technical Reference Manual. *BUT11AF datasheet*.
<https://www.mouser.com/ds/2/149/BUT11AF-888677.pdf>
- [31] BC807-25 Technical Reference Manual. *BC807-25 datasheet*.
<https://www.diodes.com/assets/Datasheets/ds11208.pdf>
- [32] PE-67100NL Technical Reference Manual. *PE-67100NL datasheet*.
<https://www.mouser.es/datasheet/2/336/-349731.pdf>